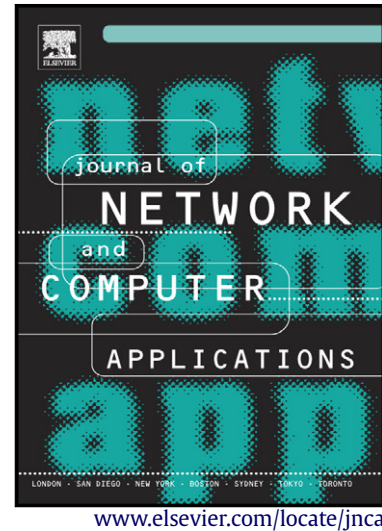


Network attacks: Taxonomy, tools and systems

N. Hoque, Monowar H. Bhuyan, R.C. Baishya, D. K. Bhattacharyya, J.K. Kalita



PII: S1084-8045(13)00175-6  
DOI: <http://dx.doi.org/10.1016/j.jnca.2013.08.001>  
Reference: YJNCA1103

To appear in: *Journal of Network and Computer Applications*

Received date: 28 January 2013

Revised date: 11 July 2013

Accepted date: 5 August 2013

Cite this article as: N. Hoque, Monowar H. Bhuyan, R.C. Baishya, D.K. Bhattacharyya, J.K. Kalita, Network attacks: Taxonomy, tools and systems, *Journal of Network and Computer Applications*, <http://dx.doi.org/10.1016/j.jnca.2013.08.001>

This is a PDF file of an unedited manuscript that has been accepted for publication. As a service to our customers we are providing this early version of the manuscript. The manuscript will undergo copyediting, typesetting, and review of the resulting galley proof before it is published in its final citable form. Please note that during the production process errors may be discovered which could affect the content, and all legal disclaimers that apply to the journal pertain.

## Network Attacks: Taxonomy, Tools and Systems

N. Hoque<sup>a,\*</sup>, Monowar H. Bhuyan<sup>a,\*</sup>, R. C. Baishya<sup>a,\*</sup>, D. K. Bhattacharyya<sup>a,\*</sup>, J. K. Kalita<sup>b,\*</sup><sup>a</sup>*Department of Computer Science & Engineering, Tezpur University  
Napaam, Tezpur-784028, Assam, India*<sup>b</sup>*Department of Computer Science, University of Colorado at Colorado Springs  
CO 80933-7150, USA*

---

**Abstract**

To prevent and defend networks from the occurrence of attacks, it is highly essential that we have a broad knowledge of existing tools and systems available in the public domain. Based on the behavior and possible impact or severity of damages, attacks are categorized into a number of distinct classes. In this survey, we provide a taxonomy of attack tools in a consistent way for the benefit of network security researchers. This paper also presents a comprehensive and structured survey of existing tools and systems that can support both attackers and network defenders. We discuss pros and cons of such tools and systems for better understanding of their capabilities. Finally, we include a list of observations and some research challenges that may help new researchers in this field based on our hands-on experience.

*Keywords:* Network attacks, tools, systems, protocol, DoS

---

**1. Introduction**

Due to the Internet's explosive growth and its all pervasive nature, users these days rely on computer networks for most day to day activities. Network attacks attempt to bypass security mechanisms of a network by exploiting the vulnerabilities of the target network. Network attacks disrupt legitimate network operations and include malfunctioning of network devices, overloading a network and denying services of a network to legitimate users, highly reducing network throughput, scanning maliciously and other similar activities. An attacker may also exploit loopholes, bugs, and misconfigurations in software services to disrupt normal network activities. Network security tools facilitate network attackers as well as network defenders in identification of network vulnerabilities and collection of network statistics. Network attackers intentionally try to identify

---

\*Corresponding authors

*Email addresses:* tonazrul@gmail.com (N. Hoque), mhb@tezu.ernet.in (Monowar H. Bhuyan), rcb@tezu.ernet.in (R. C. Baishya), dkb@tezu.ernet.in (D. K. Bhattacharyya), jkalita@uccs.edu (J. K. Kalita)

<sup>1</sup>Nazrul Hoque is a Senior Research Fellow in the Department of Computer Science and Engineering, Tezpur University, Napaam, Tezpur, Assam, India.

<sup>2</sup>Monowar Hussain Bhuyan is a PhD candidate in the Department of Computer Science and Engineering, Tezpur University, Napaam, Tezpur, Assam, India.

<sup>3</sup>Ram Charan Baishya is a Senior Research Fellow in the Department of Computer Science and Engineering, Tezpur University, Napaam, Tezpur, Assam, India.

<sup>4</sup>Dhruba Kr. Bhattacharyya is a professor in the Department of Computer Science and Engineering, Tezpur University, India.

<sup>5</sup>Jugal K. Kalita is a professor in the Department of Computer Science, University of Colorado at Colorado Springs, USA.

loopholes based on common services open on a host and gather relevant information for launching a successful attack. Thus it is of considerable interest to attackers to determine whether or not the defenders of a network are monitoring network activities regularly. Network defenders try to reduce abnormal activities from live network traffic. Defenders do not usually hide their identity during observation while attackers do.

A large number of network security tools have been designed to launch, capture, visualize, and detect different types of attacks with multiple objectives. Example tools include LOIC [1], HOIC [2], Wireshark [3], Gulp [4], Ntop [5], etc. These tools can be used for capture of live network traffic, preprocessing, feature extraction, vulnerability analysis, traffic visualization and actual detection of attacks. Thus, network security tools help in network security engineering from the viewpoint of both attackers and defenders.

### 1.1. Motivation

Even though there are several published surveys of network security tools such as [6], [7], [8], their scopes are limited and they usually discuss only a few tools. In [6], the authors discuss network attack tools which are specific to visual fingerprinting. In [7], the authors include a few tools that are commonly used by hackers. An exhaustive survey of network forensics is presented in [8]. The authors categorize the tools into two major groups, viz., network forensic analysis tools and network security tools. None of the surveys [6], [7], [8] include a taxonomy, attack launching tools, and information gathering tools. They also do not discuss recent network intrusion detection systems. Hence, in this paper we present a structured and comprehensive survey on network attacks in terms of general overview, taxonomy, tools, and systems with a discussion of challenges and observations. Our paper is detailed with ample comparisons where necessary and intended for readers who wish to begin research in this field.

### 1.2. Prior surveys

Several surveys on network security are available in the literature [6], [7], [8], [9], [10], [11], [12], [13], [14]. However, only a few surveys cover network security tools in general. Teodoro et al. [15] discuss some popular anomaly based intrusion detection techniques and systems. They focus on NIDSs and several detection techniques under three major categories, viz., statistical, knowledge based, and machine learning. Corona et al. [16] present an overview of adversarial attacks against intrusion detection systems. They provide a general taxonomy of attacks against IDSs, use of IDS weaknesses for attack implementation, and solutions for each attack they include. An overview of IDSs in terms of detection and operation is given in [17]. Debar et al. [18] present a taxonomy of IDSs from several security aspects. A few fingerprinting attack tools with their detection methodologies are briefly summarized in [6]. Out of top 75 network security tool list produced by *fyodor*, the creator of *nmap*, a few tools have been included. Barber et al. [7] present a few sophisticated attack tools with their usefulness in brief. Our survey differs from these previous surveys in view of the following points.

- (a) Unlike [19], we present tools and attack detection systems under two main categories viz., tools for network defenders and tools for attackers. Our survey also includes a comparison of tools with parameters that are useful to the network traffic analyzer.
- (b) A survey of network forensic analysis methods is reported in [8]. Similar to this survey, we describe tools for network defenders as well as tools for attackers used during network traffic capture, analysis and visualization. In addition, we also include a discussion on several IDSs, with architectures to improve the reader's understanding of the detection mechanisms.
- (c) Unlike [16], we include a taxonomy of network attacks, tools and detection systems. Tools are categorized into two classes: tools for network defenders and tools for attackers. The tools and systems are evaluated using parameters that may help on choose a tool or a system for experiment or for specific applications.

A comparison of the existing surveys on network security tools and systems is given in Table 1.

Table 1: Comparison with existing surveys

Refereces	Tools			Systems		
	Attack	Defense	Both	Host	Network	Hybrid
[6]	✓					
[7]	✓					
[8]		✓	✓	✓	✓	✓
[10]				✓	✓	
[11]				✓	✓	
[12]				✓	✓	
[13]	✓					
[18]				✓	✓	
[17]				✓	✓	✓
[19]				✓	✓	✓
[20]				✓	✓	
[21]				✓	✓	✓
[22]	✓			✓	✓	✓

### 1.3. Contributions

This paper provides a structured and comprehensive survey on network security tools and systems that are needed by network security researchers. The major contributions of this survey are the following.

1. Like the taxonomy of network security tools and systems given in [8], we classify the tools and systems into a number of categories. In addition, we also provide an analysis of tools in terms of their capabilities, performance, details of parameters and input/output.
2. Most existing surveys do not fully cover launching and information gathering tools, but we do.
3. In addition to discussing network security tools, we present a few popular NIDSs with architecture diagrams including components and functions, and also present a comparison among the NIDSs.
4. Finally, we list several important observations from both technical and practical viewpoints.

#### 1.4. Organization

The rest of the paper is organized as follows. Network attacks and related concepts are discussed in Section 2. Section 3 presents a taxonomy of network security tools, a description of these tools, in addition to comparisons among them whereas Section 4 is dedicated to attack detection systems and architectures. Finally, we present our observations and conclusions in Section 5.

## 2. Network attacks and related concepts

An attack can take many forms such as a Trojan attack, DoS/DDoS attack, or a scan attack. A DDoS attack is very catastrophic to any information system since it uses a large number of compromised hosts and it is very difficult to detect the original source of such an attack. Protecting against system compromise is a good way to defend against DDoS attacks. We discuss different types of anomalies, steps in launching attacks and their detection mechanisms.

### 2.1. Anomalies in Network

Anomalies are non-conforming interesting patterns compared to the well-defined notion of normal behavior. Traffic anomalies in computer networks are categorized as network operation anomaly, flash crowds and network abuse anomaly. All these anomalies can be detected by analyzing the traffic volume transmitted from station to station. In addition to these anomalies, other anomalies such as ALPHA, DoS/DDoS, scan, worm, outage, ingress shift, information gathering, passive attacks, spoofing attacks, man-in-middle attacks, and DNS cache poisoning, can cause damage and destruction to the network environment. Anomalies can have large impacts on both performance and security. For example, network anomalies cause service degradation and impact on network speed, as a result of which, network performance may suffer considerably.

### 2.2. Steps in Launching an Attack

Generally, an attacker executes four basic steps [23] to launch an attack. The steps are given below.

- (i) *Information Gathering*: The attacker attempts to gather vulnerability information from the network with the hope that some of the information can be used to aid in the ensuing attack.
- (ii) *Assessing Vulnerability*: Based on the vulnerabilities learned in the previous step, the attacker attempts to compromise some nodes in the network by exploiting malicious code, as a precursor to the launching of attack(s).
- (iii) *Launching Attack*: The attacker launches the attack on the target victim machine(s) using the compromised nodes.
- (iv) *Cleaning up*: Finally, the attacker attempts to eliminate the attack history by cleaning up all the registry or log files from the victim machine(s).

### 2.3. Launching and Detecting Attacks

Before launching an attack, an attacker first attempts to gather vulnerability information about the target system that may help in attack generation. An attacker scans the network using information gathering tools like nmap and finds loopholes in the system. Based on the gathered information, the attacker exploits some malicious code, possibly available on the network. The malicious code may be used to first compromise hosts in the network or it may be used to directly launch an attack and disrupt the network. There are many methods for launching an attack. For example, one may use Trojans or worms to generate an attack on a system or a network. Scanning or information gathering may be coordinated with an attack and performed simultaneously. One can also use attack launching tools such as Dsniff [24], IRPAS [25], Ettercap [26] and Libnet [27] to generate MAC attacks, ARP attacks or VLAN attacks. The main purpose of the attacker in many cases is to disrupt services provided by the network either by consuming resources or consuming bandwidth. These types of attacks can be launched using flooding of legitimate requests as in TCP SYN Flooding, ICMP flooding and UDP flooding.

To detect an attack, one must know the characteristics of an attack and its behavior in a network. The network administrator needs a visualization or monitoring system to observe differences between the characteristics of abnormal traffic and the normal. An attack can be detected from the traffic volume based on the packet header or network flow information. However, such detection usually requires processing huge volumes of data in near real-time. Obviously, designing a real-time defense mechanism that can identify all attacks is a challenging and quite likely impossible task. Most detection methods need some prior information about attack characteristics to use during the detection process. The evaluation of these intrusion detection mechanisms or systems is performed using misclassification rate or false alarm rate. To obtain satisfactory results, an IDS designer needs to be careful in choosing an approach, matching mechanism or any heuristic or in making assumptions. Approaches that have been able to obtain acceptable results include statistical [28], soft computing [29], probabilistic [30], knowledge-based [31] and hybrid [32]. A detailed discussion of these approaches is available in [33, 34].

Detection systems are designed to protect the network from different types of vulnerabilities, which may crash the network or may capture private or secure information. Deployment of an accurate and efficient anomaly detection system demands appropriate design as per standard security requirements and risk analysis. The detection system can be either host based or network based. A typical network structure with a protected LAN, a demilitarized zone and a deployed IDS console is shown in the Figure 1. A demilitarized zone (sometimes referred to as a perimeter network) is a physical or logical subnetwork that contains and exposes an organization's external-facing services to a larger untrusted network, usually the Internet. As shown in the figure, an attacker may launch an attack from various machines connected to the network either via wired or wireless media. The increasing number of highly sophisticated attacks of complex and evolving nature has made the task of defending networks challenging. The appropriate use of tools and systems can simplify the task significantly. This necessitates an awareness of the characteristics

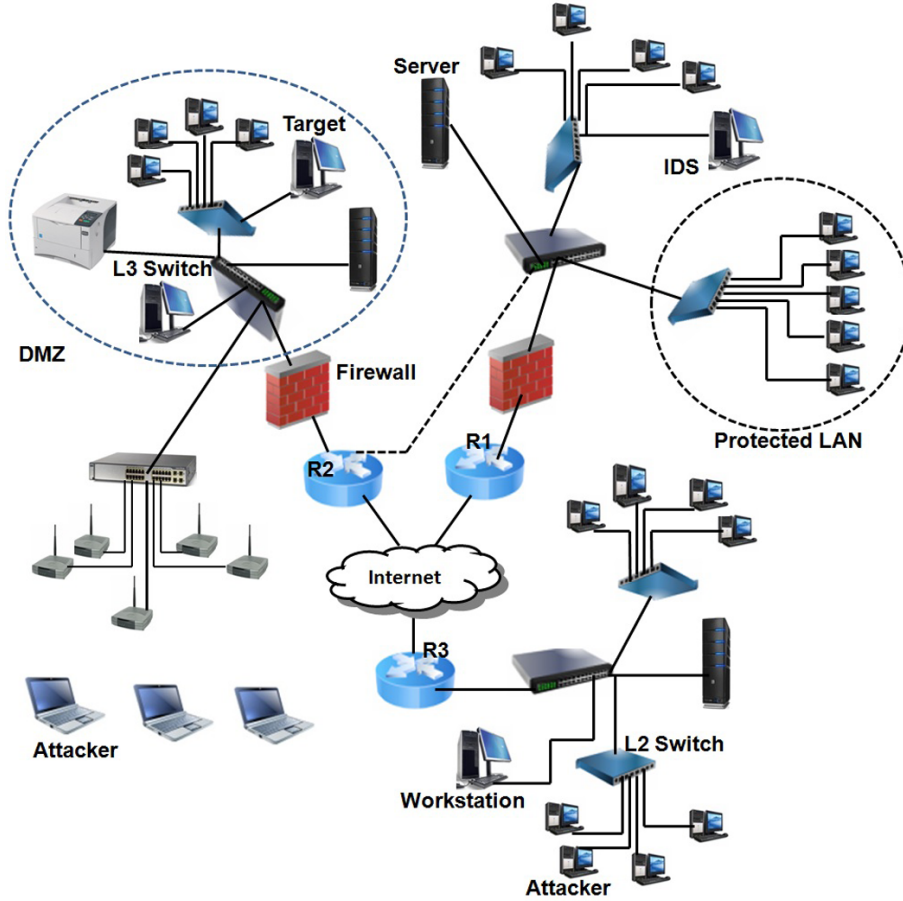


Figure 1: A typical network structure with protected LAN, DMZ and IDS deployment

and relevance of these tools and systems, and their usage.

### 3. Network Security Tools

People use different attack tools to disrupt a network for different purposes. As mentioned earlier, attackers generally target Web sites or databases as well as enterprise networks by gathering information based on their weaknesses. In general, attackers use relevant tools for the class of attack they desire to launch. A large number of defense tools also have been made available by various network security research groups as well as private security professionals. These tools have different purposes, capabilities and interfaces.

We categorize existing tools into two major categories: tools for attackers and tools for network defenders. A taxonomy of the tools used in network security is shown in Figure 2. For each basic category, we show subcategories considering their general characteristics.

#### 3.1. Information Gathering Tools

Before launching an attack, attackers need to understand the environment where the attack is to be launched. To do so, attackers first gather information about the network such as the port numbers of machines and services, operating systems, and so forth. After gathering information,

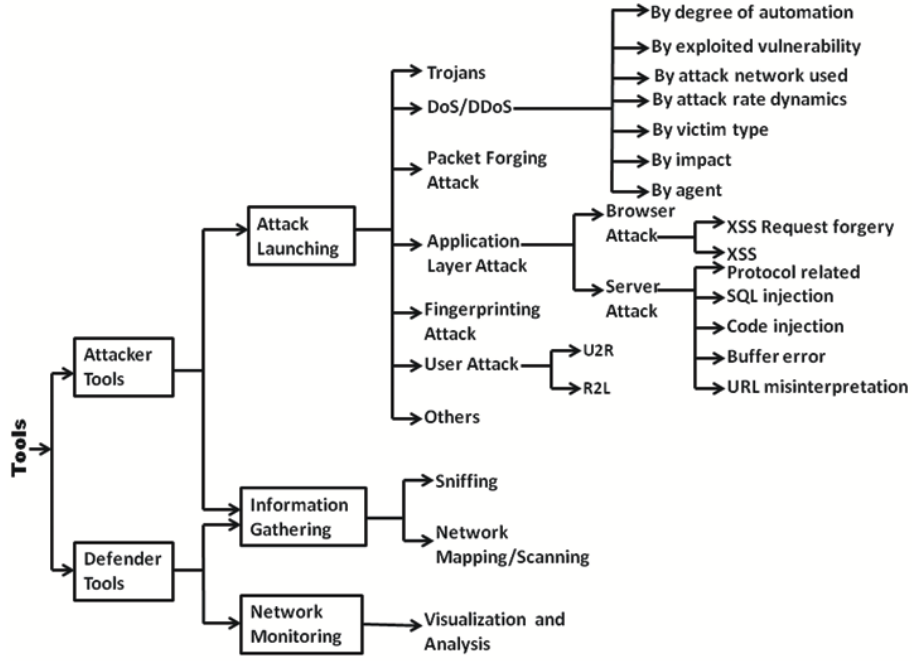


Figure 2: Taxonomy of network security tools

attackers find weaknesses in the network using various tools. Information gathering tools are further classified as sniffing tools and network mapping/scanning tools.

### 3.1.1. Sniffing Tools

A sniffing tool aims to capture, examine, analyze and visualize packets or frames traversing across the network. To support extraction of additional packet features and for their subsequent analysis, it also requires the protocol information during visualization. Some packet sniffing tools are discussed below.

- (i) *Tcpdump*: Tcpdump is a premier packet analyzer for information security professionals. It enables one to capture, save and view packet data. This tool works on most flavors of the Unix operating system. One can also use third party open source software, e.g., wireshark to open and visualize tcpdump captured traffic.
- (ii) *Ethereal*: Ethereal is a sniffing and traffic analyzing software tool for Windows, Unix and Unix-like OSs, released under the GNU license scheme. It includes two primary library utilities, (a) *GTK+*, a GUI based library, and (b) *libpcap*, a packet capture and filtering library. Ethereal is also capable of reading the output of tcpdump and can apply tcpdump filters to select and display records satisfying certain parameters. Ethereal offers decoding options for a large number ( $\geq 400$ ) of protocols and is useful in network forensics. It supports preliminary inspection of attacks in the network.
- (iii) *Net2pcap*: It is a simple tool to read packet traffic from an interface and to transform into a pcap file. Net2pcap is a Linux tool which does not use any library during the transformation.



However, it is partially dependent on *libc*, a Linux library utility. The command

```
%tcpdump -w capfile
```

almost does the same task as Net2pcap. However, Net2pcap is usually used to capture and represent network traffic in a hostile environment to support subsequent analysis.

- (iv) *Snoop*: It is a Linux based tool, that works like tcpdump. However, the format of a snoop file is different from the *pcap* format and is defined in RFC 1761<sup>1</sup>. An important feature of this tool is that when writing to an intermediate file, it reduces the possibility of packet loss under busy trace conditions. Snoop allows one to filter, read as well as to interpret packet data. To observe the traffic between two systems, say *X* and *Y*, the following command is used to execute the tool.

```
% snoop X, Y
```

- (v) *Angst*: Angst runs on Linux and OpenBSD and is an active packet sniffer that can capture data on switched networks by injecting data into the network. Angst is able to flood a network using random MAC addresses, by causing switches to transmit packets towards all ports.
- (vi) *Ngrep*: Ngrep provides filtering facility on packet payloads. It also supports sniffing with the help of tcpdump and libpcap.
- (vii) *Ettercap*: Ettercap is a very good sniffer that runs on almost all platforms. More of an active hacking tool, Ettercap uses an ncurses interface and is able to decode several protocols. Ettercap operates in multipurpose mode: sniffer, and interceptor or logger mode for switched LANs. Ettercap can collect passwords for multiple applications, kill connections, inject packets, inject commands into active connections, and has additional plugins.
- (viii) *Dsniff*: Dsniff is a collection of tools that enable active sniffing on a network. It can perform man-in-the-middle attacks against SSHv1 and HTTPS sessions. It can also sniff switched networks by actively injecting data into the network and redirecting traffic.
- (ix) *Cain & Able*: It is a multipurpose sniffer tool that runs on Windows NT, 2000 and XP and allows for password recovery for a number of protocols, including MSN messenger, and RADIUS shared keys. It can also launch man-in-the-middle attacks for SSHv1 traffic.
- (x) *Aimsniff*: It is a simple tool to capture the IP address of an AOL Instant Messenger user while a direct connection is established between the user with others. Once the connection is established, one is able to simply click on the sniff button to capture the IP address.
- (xi) *Tcptrace* : It is a powerful tool to analyze tcpdump files and to generate various types of outputs including connection specific information, such as the number of bytes and segments

---

<sup>1</sup><http://snoopwpf.codeplex.com/>

sent and received, elapsed time, retransmissions, round trip times, window advertisements and throughput. It accepts a wide range of input files generated by several capture tools such as tcpdump, snoop, etherpeek, HP Net Metrix, and WinDump. It also provides a graphical presentation of traffic characteristics for further analysis.

- (xii) *Tcptrack*: It can sniff and display TCP connection information, as seen in the network interface. Tcptrack can perform the following functions: (a) watch passively for connections on the network interface, (b) keep track of their state and (c) display a list of connections. It displays source IP, destination IP, source port, destination port, connection state, idle time, and bandwidth usage.
- (xiii) *Nstreams*: It is a tool to display and analyze network streams generated by users between several networks, and between networks and the outside. Nstreams also can output optionally the *ipchains* or *ipfw rules* matching these streams. It parses outputs generated by tcpdump or files generated using tcpdump with *-w* option.
- (xiv) *Argus*: Argus runs on several operating systems such as Linux, Solaris, Mac OS X, FreeBSD, OpenBSD, NetBSD, AIX, IRIX, Windows and OpenWrt. It can process either live traffic or captured traffic files and can output status reports on flows detected in the stream of packets. It reports reflect flow semantics. This tool provides information on almost all packet parameters, such as reachability, availability, connectivity, duration, rate, load, loss, jitter, retransmission, and delay metrics for all network flows.
- (xv) *Karpski*: This is a user-friendly tool with limited sniffing and scanning capabilities. It provides flexibility to include protocol definitions dynamically and also can serve as an attack launching tool against addresses on a local network.
- (xvi) *IPgrab*: This packet sniffing tool provides facility for network debugging at multiple layers, such as data link, network and transport layers. It outputs detailed header field information for all layers.
- (xvii) *Nast*: Nast uses libnet and libpcap to sniff packets in normal mode or in promiscuous mode to analyze them. It captures packet header parameters, payload information, and saves them in a file in ASCII or ASCII-hex format.
- (xviii) *Aldebaran*: It is an advanced libpcap-based sniffing and filtering tool for the TCP protocol. It provides basic information about the source and destination addresses and ports, but no information regarding flags. One can use it to monitor data sent by connections as well as to sniff passwords. Based on libpcap rules, one can use it to sniff packet headers as well as payload contents, and can transmit captured traffic to another host via UDP. Aldebaran also allows one (a) to encrypt the contents saved in dump files, (b) to analyze interface traffic and (c) to report packet statistics, viz., packet count, size, and average speed in HTML or as a plain text file.

- (xix) *ScoopLM*: This is a Windows 2000 based sniffing tool for capturing LM/NTLM authentication information on the network. Such information can later be used by a tool such as BeatLM to crack authentication data.
- (xx) *Gulp*: It is used to capture very high volume network traffic efficiently from the network firehose. It overcomes the packet loss problem of tcpdump and records a large amount of data that are stored in secondary storage for further processing. It can capture packets from multiple CPUs for better performance and writes the captured data in pcap files.
- (xxi) *Nfsen*: Nfsen is used to visualize NetFlow flow data and allows one to display the captured data such as flows, packets and bytes using a graphical interface. One can also visualize protocol specific flows in a graphical format using Nfsen.
- (xxii) *Nfdump*: It is used to collect and process NetFlow data. It reads NetFlow data from the files created by nfcapd. It organizes captured data in a time based fashion, typically every 5 minutes and stores them for further processing. Analysis of data can be performed for a single file, or by concatenating several files in a single run. The output is either ASCII text or binary data, when saved into a file, ready to be processed again with the same tool.

Table 2: Comparison of sniffing tools

Tool's name	Input	Protocol	Purpose	Effectiveness	Sources
Ethereal	I	C/HT/S	packet capturing	powerful, user friendly	www.ethereal.com.
Tcpdump	I	C/U/IC	packet capturing	less intrusive than Ethereal	www.tcpdump.org
Net2pcap	I	C/U/IC	packet capturing	Linux based, auditable	www.secdev.org
Snoop	N	C/U/IC/Te/ F	packet capturing	no packet loss, supports more than 12 options	www.softpanorama.org
Angst	H/p	C	sniffing	easy to use, aggressive	www.angst.sourceforge.net
Ngrep	I	C/U/IC	capturing packet	multi-platform, handles large data	www.ngrep.sourceforge.net
Ettercap	I	C/U	man-in-the-middle attack	efficient, supports more than 35 options	www.ettercap.sourceforge.net
Dsniff	I	F/Te/S/HT/P	password sniffing	Unix based	www.naughty.monkey.org
Cain & able	I		password recovery	easy to use	www.oxid.it
Aimsniff	H	C/U/HT	capturing packet	Linux based	www.sourceforge.net
Tcptrace	F	C	analysis of traffic	most commonly used	www.tcptrace.org
Tcptrack	I/p	C	TCP connection analysis	Linux based	www.rhythm.cx
Argus	F	C/U	analysis of audit data	multi-platform, real-time processing	www.qosient.com/argus
Karpski	I	C/U	packet analyzer	limited applicability	www.softlist.net
IPgrab	I/p		display packet header	displays packet details	www.ipgrab.sourceforge.net
Nast	I	C/U	traffic analysis	supports more than 12 options	www.nast.berlios.de
Gulp	I	C/U/IC	packet capturing/ visualization	very efficient, easy to use	staff.washington.edu/corey
Libpcap	I	C/U/IC	packet capturing	high performance	www.tcpdump.org
Nfsen	I	C/U	flow capturing/ visualization	easy navigation of NetFlow data	www.nfsen.sourceforge.net
Nfdump	I	C/U	flow capturing/ visualization	powerful packet analyzer	www.nfdump.sourceforge.net

Here, I-Interface ID, N-Network IP, H-Host IP, F-Traffic captured file, p-port, C-TCP, U-UDP, IC=-CMP, IG-IGMP, HT-HTTP, S-SMTP, F-FTP, P-POP, Te-Telnet

We note that sniffing tools we have discussed above are not equally useful for all purposes all the time. Their usefulness and importance depend on the user's requirements and purpose at a certain point in time. For example, one cannot use the Cain & Able to capture live network

traffic since it performs only password cracking. Most people use tcpdump and libpcap as network sniffing tools to capture all information in packets and store them in a file. One can use the Nfsen and Nfdump tools for NetFlow traffic capture whereas Gulp is used for packet level traffic capture. However, these tools also use tcpdump as an implicit tool for packet as well as NetFlow capture.

### 3.1.2. Scanning tools

A network scanning tool aims to identify active hosts on a network, either (a) to attack them, or (b) to assess vulnerabilities in the network. It provides an overall status report regarding network hosts, ports, IPs, etc. The four possible types of port scans are (i) one-to-one, (ii) one-to-many, (iii) many-to-one, and (iv) many-to-many as shown in Figure 3. Below, we present a few network vulnerability scanning tools.

- (i) *Nmap*: This network mapping tool facilitates network exploration and security auditing. It can scan large networks fast, especially against single hosts. It is effective in using raw IP packets to identify a large number of useful parameters, such as available hosts, services offered by the hosts, OSs running, and use of packet filters or firewalls. In addition to its use in security audits, network administrators can use it for routine tasks such as maintaining network inventory, managing service upgrade schedules, and monitoring host or service uptime.
- (ii) *Amap*: Amap detects an application protocol without depending on the TCP/UDP ports it is bound to. It identifies applications running on a specific port by sending trigger packets, which are typically involved in an application protocol handshake. Most network daemons only respond to the correct handshake (e.g., SSL). Amap takes the responses and looks for matches. This tool supports TCP and UDP protocols, regular and SSL-enabled ASCII and binary protocols and has a wide list of options to control its behavior. It accepts an nmap machine readable output file and logs to a file and a terminal.
- (iii) *Vmap*: This version mapper tool allows one to identify the version of a daemon by fingerprinting its characteristics, based on its responses to bogus commands.
- (iv) *Unicornscan*: This is an asynchronous scanner as well as a payload sender. This scalable and flexible tool gathers and collects information quickly. For fast response, it uses a distributed TCP/IP stack and provides a user-friendly interface to introduce a stimulus into a TCP/IP enabled device or network, and measure the response. The main features of this tool include asynchronous protocol specific UDP scanning, asynchronous stateless TCP scanning with wide variations in TCP flags and asynchronous stateless TCP banner grabbing.
- (v) *TTLscan*: This tool uses libnet and libpcap utilities to identify a host by sending TCP SYN packets to each port of the host. It sniffs the response from the host and uses it to identify hosts with services by forwarding packets to another host behind a firewall. It can detect

the OS and its version running on a host behind the firewall by reading specific header parameters such as TTL, window size and IP ID.

- (vi) *IKE-scan*: This tool assists in discovery, fingerprinting and testing of IPSec VPN servers based on the IKE protocol. IKE-scan works on Linux, Unix, Mac OS and Windows environment under the GPL license.
- (vii) *Paketto*: It is a set of tools to assist in manipulating TCP/IP networks based on non-traditional strategies. These tools provide tapping functionality within the existing infrastructure and also extend protocols beyond their original purpose. Example tools include (a) *scanrand*, which facilitates fast discovery of network services and topologies, (b) *minewt*, which serves as a user space NAT/MAT router, (c) *linkcat*, which offers an Ethernet link to stdio, (d) *paratrace*, which helps trace network paths without spawning new connections, and (e) *phentropy*, which uses *OpenQVIS* to render arbitrary amounts of entropy from data sources in 3-D phase space.

Table 3 shows the purposes and effectiveness of these tools and the sources from where they can be obtained. Almost all these tools are Linux based.

Table 3: Comparison of scanning tools

Tool's name	Input	Protocol	Purpose	Effectiveness	Sources
Nmap	IP/p	C/U	scanning	very powerful, easy to use	www.insecure.org
Amap	IP/p	C/U	scanning	powerful application mapper	www.freeworld.thc.org
Vmap	T	C/U	version mapping	few options, easy to use	www.tools.l0t3k.net
Unicornscan	T/p	C/U	scanning	supports more than 15 options	www.unicornscan.org
Ttlscan	T/p	C	scanning	Linux based	www.freebsd.org
Ike-scan	T	C/U	host discovery	supports more than 50 options	www.stearns.org
Paketto	IN	C	scanning	very fast scanner	www.packages.com

Here, IP-IP address(es), T-Target IP, p-port, IN-Interface ID, C-TCP, U-UDP

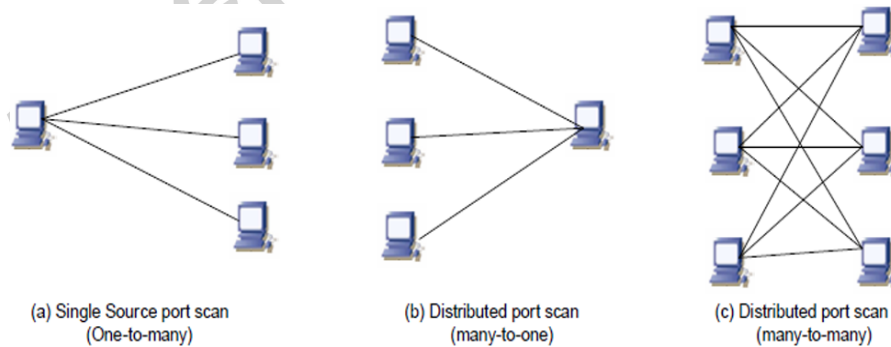


Figure 3: Different types of port scans

For scanning a large network, one can use nmap as the most effective tool. Nmap has the ability to scan a large network to determine multiple parameters such as active hosts and ports, host operating systems, protocols, timing and performance, firewall/IDS evaluation and spoofing, and

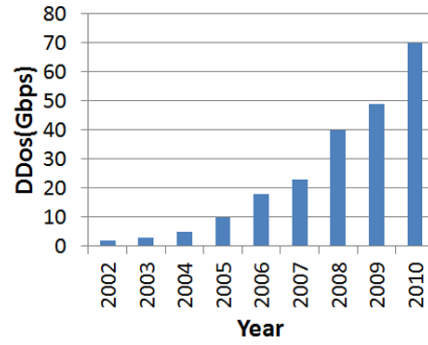


Figure 4: DDoS attack graph

IPv6 scanning. Due to its multiple functionalities, network administrators find it very useful to monitor a large network. Amap and Vmap do not support many of the functionalities performed by nmap. Attackers use namp to find the vulnerabilities in a host to compromise it for constructing BotNets during DDoS attack generation using the agent handler architecture.

### 3.2. Attack Launching Tools

A large number of network security tools that use cryptographic mechanisms to launch attacks are available on the Web. People can freely download these tools and can use them for malicious activities such as Trojan propagation, network mapping, probe attacks, buffer overflow attacks, DoS/DDoS attacks, and application layer attacks. Such tools can be used to launch layer specific and protocol specific attacks, such as HTTP, SMTP, FTP or SNMP related attacks. Other tools can be used to launch DoS/DDoS attacks, which can disrupt the services of a network or a Website very quickly. Some tools are used in wired networks to capture and exploit valuable information while others are used in wireless networks.

#### 3.2.1. Trojans

Trojans are malicious executable programs developed to break the security system of a computer or a network. A Trojan resides in a system as a benign program file. Once the user attempts to open the file, the Trojan is executed, and some dangerous action is performed. Victims generally unknowingly download the Trojan from multiple sources such as (i) the Internet, (ii) an FTP archive, (iii) via peer-to-peer file exchange using IRC, and (iv) Internet messaging. Typically, Trojans are of seven distinct types: (a) Remote access Trojans (b) Sending Trojans (c) Destructive Trojans (d) Proxy Trojans (e) FTP Trojans (f) Security software disable Trojans and (g) DoS Trojans.

*Remote access Trojans* are malware programs that use backdoors to control the target machine with administrative privilege. These type of Trojans are downloaded invisibly with a user request for a program such as a game or an email attachment. Once the attacker compromises a machine, the Trojan uses this machine to compromise more machines to construct a BotNet for launching a DoS or DDoS attack. An example of remote access Trojan is *danger*. *Sending Trojans* are

used to capture and provide sensitive information such as passwords, credit card information, log files, e-mail addresses, and IM contact lists to the attacker. In order to collect such information, such Trojans attempt to install a keylogger to capture and transmit all recorded keystrokes to the attacker. Examples of this type of Trojans are *Badtrans.B email virus*, and *Eblast*. *Destructive Trojans* are very destructive for a computer and often programmed to delete automatically some essential executable programs such as configuration and dynamic link library (DLL) files. Such Trojans act either (i) as per the instructions of a back-end server, or (ii) based on pre-installed or programmed instructions, to strike on a specific day, at a specific time. Two common examples of this type are *Bugbear virus* and *Goner worm*. *Proxy Trojans* attempt to use a victim's computer as a proxy server. A Trojan of this kind compromises a computer and attempts to perform malicious activities such as fraudulent credit card transactions, and launching of malicious attacks against other networks. Examples of proxy Trojans are *TrojanProxy.Win32*, *Paramo.F*. *FTP Trojans* attempt to open port 21 and establish a connection from the victim computer to the attacker using the File Transfer Protocol (FTP). An example of FTP Trojan is *FTP99cmp*. *Security software disable Trojans* attempt to destroy or to thwart defense mechanisms or protection programs such as antivirus programs or firewalls. Often such a Trojan is combined with another type of Trojan as a payload. Some examples are *trojan.Win32.KillAV.ctp* and *trojan.Win32.Disable.b*. *DoS Trojans* attempt to flood a network instantly with useless traffic, so that it cannot provide any service. Some examples of this category of Trojan are *ping of Death*, and *teardrop*.

### 3.2.2. DoS/DDoS Attacks

*Denial of service* (DoS) is a commonly found, yet serious class of attack caused due to an explicit attempt of an attacker to prevent or block legitimate users of a service from using desired resources. Such an attack occurs in both distributed as well as in a centralized setting. Some common examples of this class of attack are: *SYN flooding*, *smurf*, *fraggle*, *jolt*, *land*, and *ping-of-death*.

A *Distributed Denial of Service* (DDoS) attack is a coordinated attempt on the availability of services of a victim system or a group of systems or on network resources, launched indirectly from a large number of compromised machines on the Internet. Typically, a DDoS attacker adopts an  $m : 1$ , i.e., many compromised machines to a single victim machine or an  $m : n$  approach that makes it very difficult to detect or prevent. A DDoS attacker normally initiates such a coordinated attack using either an architecture based on agent handlers or Internet relay chat (IRC). The attacking hosts are usually personal computers with broadband connections to the Internet. These computers are compromised by viruses or Trojan programs called *bots*. These compromised computers are usually referred to as *zombies*. The actions of these zombies are controlled by remote perpetrators often through (a) *BotNet* commands and (b) a control channel such as IRC. Generally, a DDoS attack can be launched using any one of the following ways.

- (i) *By degree of automation*: The attack generation steps such as recruit, exploit, infect, and use phase can be performed in three possible ways: manual, automatic, and semi-automatic.

- (ii) *By exploited vulnerability*: The attacker exploits the vulnerability of a security system to deny the services provided by that system to legitimate users. In semantic attacks, it exploits a specific feature or implementation bug of some protocols or applications installed in the victim machine to overload the resources used by that machine. An example of such attack is the TCP SYN attack.
- (iii) *By attack network used*: To launch a DDoS attack, an attacker may use either an agent handler network or an IRC network.
- (iv) *By attack rate dynamics*: Depending on the number of agents used to generate a DDoS attack, the attack rate may be either a constant rate or a variable rate attack. Besides these, an increasing rate attack and a fluctuating rate attack can also be mounted using a rate change mechanism.
- (v) *By victim type*: DDoS attacks can be generated to paralyze different types of victims. Example include application attacks, host attacks, network attacks, and infrastructure attacks.
- (vi) *By impact*: Based on the impact of a DDoS attack, it may be either a disruptive or a degrading attack.
- (vii) *By agent*: A DDoS attack can be generated by a constant agent set or a variable agent set.

Some statistics on DDoS attacks are shown in Figure 4. Out of many DoS/DDoS attack generation tools, a few are discussed below.

- (i) *Jolt*: This DoS attack tool sends a large number of fragmented ICMP packets to a target machine running Windows 95 or NT in such a manner that the target machine fails to reassemble them for use, and as a result, it freezes up and cannot accept any input from the keyboard or mouse. However, this attack does not cause any significant damage to the victim system, and the machine can be recovered with a simple reboot.
- (ii) *Burbonic*: This DoS exploit attempts to victimize a Windows 2000 machine by sending a randomly large number of TCP packets with random settings with the purpose of increasing the load on the machines so that it leads to a crash.
- (iii) *Targa*: Targa is a collection of 16 different DoS attack programs. One can launch these attacks individually as well as in a group and can damage a network instantly.
- (iv) *Blast20*: This TCP service stress tool is able to identify potential weaknesses in the network servers quickly. An example use of this tool is shown below.

```
% blast targetIP port start_size end_size /b (i.e. begin text) "GET/SOME TEXT " /e (i.e. end text) "URL"
```

The command is used to send attack packets of size minimum *start\_size* bytes to maximum *end\_size* bytes to a server with a specified target IP.



- (v) *Crazy Pinger*: It attempts to launch an attack by sending a large number of ICMP packets to a victim machine or to a large remote network.
- (vi) *UDPFlood*: This tool can flood a specific IP at a specific port instantly with UDP packets. The flooding rate, the maximum duration and the maximum number of packets can be specified in this tool. It can also be used to test the performance of a server.
- (vii) *FSMax*: It is a server stress testing tool. To test a server for buffer overflows that may be exploited during an attack, it accepts a text file as input and executes a server through a sequence of tests based on the input.
- (viii) *Nemsey*: The presence of this tool implies that a computer is insecure, and infected with malicious software. It attempts to launch an attack with a specified number of packets of specified sizes, including information such as protocol and port.
- (ix) *Panther*: This UDP based DoS attack tool can flood a specified IP at a specified port instantly.
- (x) *Slowloris*: It creates a large number of connections to a target victim Web server by sending partial requests, and attempts to hold them open for a long duration. As a consequence, the victim servers maintain these connections as open, consuming their maximum concurrent connection pool, which eventually compels them to deny additional legitimate connection attempts from clients.
- (xi) *BlackEnergy*: This Web based DDoS attack tool, an HTTP-based BotNet, uses IRC based command and control method.
- (xii) *HOIC*: It is an HTTP based DDoS tool that focuses on creation of high speed multi-threads to generate HTTP flood traffic. It is able to flood simultaneously up to 256 Web sites. The built-in scripting system in this tool allows the attacker to deploy boosters, which are scripts designed to thwart DDoS counter measures.
- (xiii) *Trinoo*: This is an effective DDoS attack tool, that uses a master host and several broadcast hosts. It issues commands using TCP connection to the master host, and the master instructs the broadcast hosts via UDP to flood specific target IPs at random ports with UDP packets. To launch an attack using this tool, an attacker should possess prior access to the host to install a Trinoo master or broadcast, either by passing or by compromising the existing security system.
- (xiv) *Shaft*: It is a variant of Trinoo, which provides statistics on TCP, UDP and ICMP flooding attacks. These help attackers identify the victim machine's status (i.e., completely down or alive), or to decide on the termination of zombies in addition to the attack.
- (xv) *Knight*: This IRC-based tool can launch multiple DDoS attacks to create SYN flood, UDP flood, and urgent pointer flood on Windows machines.

- (xvi) *Kaiten*: This is an IRC-based attack tool and is able to launch multiple attacks, viz., UDP flood, TCP flood, SYN flood, PUSH+SYN flood attacks. It uses random source addresses.
- (xvii) *RefRef*: RefRef is used to exploit existing SQL injection vulnerabilities using features included in MySql such as SELECT permissions to create a DoS attack on the associated SQL server. It sends malformed SQL queries carrying payloads which force servers to exhaust their own resources. It works with a Perl compiler to launch an attack.
- (xviii) *LOIC*: It is an anonymous attacking tool launched via IRC. It operates in three modes based on the three protocols: TCP, UDP, and HTTP. It exists in two versions: binary and Web-based. It uses multiple threads to launch an attack.
- (xix) *Hgod*: This is a Windows XP based tool that spoofs source IPs and specifies protocols and port numbers during an attack. By default, it is used for TCP SYN flooding. An example of TCP SYN flooding attack command against 192.168.10.10 on port 80 with a spoofed address of 192.168.10.9, is shown below.  
`%hgod 192.168.10.10 80 -s 192.168.10`
- (xx) *TFN*: Like Trinoo, TFN requires a client host and several daemon hosts. It is very effective in launching DDoS attacks, viz., ICMP flood, UDP flood, SYN flood, and smurf attacks. TFN2K, a variant of TFN, also includes some special features, such as encryption and decryption, the ability to launch stealth attacks, and DoS attacks to crash a specified target host, and to communicate shell commands to the daemons.
- (xxi) *Stacheldrath*: This DDoS attacking tool is a hybridization of TFN and Trinoo, with some additional features, such as encrypted transmission between the components, and automatic updation of the daemons.

A large number of attack generation tools are available on the Internet and most are very powerful, and can be easily used to crash networks and Websites. Among these, we found that LOIC and HOIC are very adept at launching DDoS attacks within a very short time. LOIC supports TCP, UDP, and HTTP protocols to construct attack packets whereas HOIC supports only the HTTP protocol. Although TFN, Trinoo, and Stachaldraht can be used to launch a DDoS attack, they are not as powerful as LOIC. It should be noted that the use of LOIC to launch an attack in a public network is a crime.

### 3.2.3. Packet Forging Attack Tools

Packet forging tools are useful in forging or manipulating packet information. An attacker can generate traffic with manipulated IP addresses based on this category of tools. We describe some commonly used packet forging tools.

- (i) *Packeth*: Packeth is a Linux based tool with a graphical user interface. It can send any packet or sequence of packets using raw sockets on the Ethernet. It provides a large number of options such as being able to create incorrect checksum, and wrong header length.

Table 4: Comparison of attacking tools

Tool's name	Input	Protocol	Purpose	Effectiveness	Sources
Jolt	T	IC	DoS	uses 100% CPU time	www.flylib.com
Burbonic	T	C	DoS	multi-platform, easy to use	www.packetstormsecurity.org
Targa	T	C/U/IC	DoS	very efficient	www.packetstormsecurity.org
Blas20	T	C	DoS	multi-platform, performs quick damage to a system	
Crazy Pinger	V	IC	DoS	multi-platform, easy to use	www.softwaretopic.informer.com
UDPFlood	V	U	DoS	Windows based, less powerful	www.foundstone.com
FSMax	F		DoS	Windows based, efficient for server testing	www.brothersoft.com
Nemsey	T/p	C	DoS	Windows based	packetstormsecurity.org
Panther	T/p	U	DoS	easy to use	www.bestspywarescanner.net
Land & LaTierra	T	C	DoS	powerful for land attack	
Slowloris	T	HT	DoS	powerful for HTTP attack	www.hackers.org/slowloris
Blackenergy	S	C/U/IC	DDoS	simple and powerful for DDoS	www.airdemon.net
HOIC	T	HT	DDoS	very effective for DDoS	www.rapidshare.com
Shaft	V	U/C/IC	DDoS	multi-platform, commonly used	
Knight	V	C/U	DDoS	less powerful	www.cert.org
Kaiten	V	U/C	DDoS	Windows based	www.mcafee.com
RefRef	T		DDoS	effective for DDoS	www.hackingalert.net
Hgod	T/p	C/U/IC	DDoS	easy to use	www.flylib.com
LOIC	T/p	C/U/IC	DDoS	very effective, powerful for flooding attack	www.sourceforge.net
Trinoo	T/p	U	DDoS	multi-platform, easy to use	www.nanog.org
TFN	T	U/C/IC	DDoS	multi-platform, effective for flooding attacks	www.codeforge.com
TFN2K	T	U/C/IC	DDoS	simple and easy to execute	www.goitworld.com
Stachaldraht	T	C	DDoS	multi-platform, supports more features	www.packetstormsecurity.org
Mstream	T	C	DDoS	multi-platform and more primitive	www.ks.uiuc.edu
Trinity	T	C/U	DDoS	very effective to compromise hosts	www.garykessler.net
Here, T-Target IP, V-Victim IP, S-Server IP, C-TCP, U-UDP, IC-ICMP, F-Input text file, p-Port, HT-HTTP					

- (ii) *Packetit*: This network auditing tool allows one to customize, inject into, monitor, and manipulate IP traffic. It can be used in various ways such as (a) to test an NIDS, (b) to evaluate the performance of a firewall, (c) to scan a network, (d) to simulate network traffic, and (e) in TCP/IP auditing.
- (iii) *Packet excalibur*: This forging tool allows one to (a) sniff packets, (b) build and receive custom packets, and (c) to spoof packets. It has a graphical interface to help a user build scripts as a text file and to specify additional protocols.
- (iv) *Nemesis*: This Unix-like and Windows based network packet crafting and injection tool is useful for testing any NIDS, firewall or IP stack and a variety of other tasks. This command-line driven tool also provides an option for scripting. Nemesis allows an attacker to craft and inject a large variety of packets. Especially in IP and the Ethernet injection modes, it allows one to craft and inject almost any custom packets.
- (v) *Tcpinject*: This forging tool allows one to transmit a wide variety of TCP/IP packets by specifying multiple parameters such as source IPs, destination IPs, source ports, destination ports, packet size, payload, TCP control flags and TCP window size.
- (vi) *Libnet*: This tool provides many facilities to the application programmer including the ability to construct and inject network packets through a portable and high-level API. To

support underlying packet creation and injection functionality, it uses the libnet utility.

- (vii) *SendIP*: This command-line forging tool allows one to send arbitrary IP packets with a large number of options to specify the content of every header of a specific packet. Any data can be added to the packet during transmission.
- (viii) *IPsorcery*: This TCP/IP packet generating tool has the ability to send TCP, UDP and ICMP packets using a GTK+ interface.
- (ix) *Pacgen*: This Linux based Ethernet IP TCP/UDP packet generating tool allows an attacker to generate custom packets with configurable Ethernet, IP, TCP and UDP layers as well as custom payloads. It also includes additional features such as the ability to generate a *packet count*, and a *programmable time interval* between packets sent.
- (x) *ARP-SK*: ARP Swiss Knife (ARP-SK) allows one to create totally arbitrary ARP requests to manipulate ARP packets, and to test network security and connectivity.
- (xi) *ARPspoof*: This tool is also known as ARP Cache Poisoning. It allows one to spoof the contents of an ARP table on a remote computer on the LAN. Two addresses are used to establish connection between two computers on an IP/Ethernet network: (a) the MAC address, which is used on a local area network before packets go out of the gateway, and (b) the IP address, which is used to surf the Internet through a gateway.
- (xii) *Libpal*: This user-friendly packet assembly library provides utilities to build and send forged Ethernet, IP, ICMP, TCP and UDP packets. It uses a structure to represent a packet.
- (xiii) *Aicmpsend*: This ICMP packet sending tool supports several features including ICMP flooding and spoofing. It allows one to implement all the ICMP flags and codes.

Based on our study of packet forging tools, we note that Nemesis is widely used to generate custom packets using different protocols. It supports most protocols such as ARP, DNS, ICMP, IGMP, IP, OSPF, RIP, TCP and UDP. This makes it very effective compared to other tools. Other advantages of this tool are that: (a) anyone can generate custom packets from the command prompt or using shell scripts in a system, and (b) attackers find it very useful to generate attack packets.

#### 3.2.4. Application Layer Attack Tools

In an application layer attack, the attacker uses legitimate application layer HTTP requests from legitimately connected network machines to overwhelm a Web server [35]. The application layer attack may generate a session flooding attack, request a flooding attack or an asymmetric attack [36, 37]. Application layer DDoS attacks are more subtle than network layer attacks and the detection of application layer attacks is difficult because they use legitimate protocols and legitimate connections. Application layer attacks are of four types given below.

Table 5: Comparison of packet forging attack tools

Tool's name	Input	Protocol	Purpose	Effectiveness	Sources
Packeth	S/D	U/C/IC	packet generator	supports many features	www.sourceforge.net
Packetit	I	C/U/IC	packet analysis and injection	supports more than 60 options	www.packetfactory.openwall.net
Packet Ex-calibur				multi-platform	www.freecode.com
Nemesis	H	IC/IG/C/U	packet crafting/injection	multi-platform, powerful	www.sourceforge.net
Tcpinject	H	C	packet generator	Linux based, easy to use	www.packetstormsecurity.org
Libnet	H	C	packet injection	portable, efficient and easy to use	www.packetfactory.net/libnet
SendIP	H	C/U	packet generator	supports many options	www.softpedia.com
IPsocery	H	C/U/IC/IG	packet generator	easy to use	www.tools.l0t3k.net
Pacgen	H	C/U	packet generator	simple packet generator	www.sourceforge.net
Arp-sk	H	A	ARP packet generator	sensitive for ARP attack	www.arp-sk.org
ARP-SPOOF	T	A	packet intercept	efficient for ARP cache poisoning	www.sourceforge.net
Libpal	H	C/U/IC	packet intercept	user friendly	www.sourceforge.net
Aicmspend	T	IC	ICMP packet flooding	supports many features	www.packetstormsecurity.nl
Here, S-Source IP, D-Destination IP, I-Interface ID, H-Host IP, T-Target IP, C-TCP, U-UDP, IC-ICMP, IG-IGMP, A-ARP					

- (i) *HTTP-related attacks*: In this attack, a massive amount of HTTP requests are sent to overwhelm the target site in a very short time frame. Some commonly used tools of this category are Code Red Worm and its mutations, Nimda Worm and its mutations, Cross site scripting attacks, Malicious URLs and AppDDoS.
- (ii) *SMTP-related attacks*: SMTP protocol is used to transmit emails over Internet. The attacker tries to attack a mail server using this Internet mail transfer protocol. Some example attack tools of this category are SMTP mail flooding, SMTP worms and their mutations, extended relay attacks, and firewall traversal attacks.
- (iii) *FTP-related attacks*: The first step in this attack is to initiate a legitimate FTP connection and then send some attack packets to the victim. Examples include FTP bounce attacks, FTP port injection attacks, passive FTP attacks, and TCP segmentation attacks.
- (iv) *SNMP-related attacks*: The main goal of an SNMP attack is to change the configuration of a system and then monitor the state of availability of the system. Examples of this category of attacks include SNMP flooding attacks, default community attacks, and SNMP put attacks.

### 3.2.5. Fingerprinting Attack Tools

Fingerprinting tools are used to identify specific features of a network protocol implementation by analyzing its input and output behavior. The identified features include protocol version, vendor information and configurable parameters. Fingerprinting tools are used to identify the operating system running on a remote machine and can also be used for other purposes. Existing

fingerprinting tools show that implementations of most key Internet protocols such as ICMP, TCP, TELNET and HTTP [38, 39, 40] have bugs. Network administrators can use remote fingerprinting to collect information to facilitate management, and an intrusion detection system can capture the abnormal behavior of attackers or worms by analyzing their fingerprints [41].

- (i) *Nmap*: Nmap is one of the best fingerprinting tools for both Unix and Windows operating systems. It is very useful in network mapping as well as information gathering from a remote machine on a network as described in Subsection 3.1.2.
- (ii) *P0f*: P0f is an OS fingerprinting tool that uses passive fingerprinting in contrast to active fingerprinting performed by Nmap. Passive fingerprinting simply sniffs the network and classifies the host based on the observed traffic. This is more difficult than active fingerprinting, since one has to accept whatever communication happens rather than designing custom probes.
- (iii) *Xprobe*: This OS fingerprinting tool is used to find the operating system run by a remote machine. Xprobe is similar to Nmap and it exploits the ICMP protocol in its fingerprinting approach.
- (iv) *CronOS*: This fingerprinting tool is used to determine the operating system of a target machine. This tool is embedded in Namp-CronOS and it has three options to perform operations. The *S* option guesses the timeout period of SYN\_RCVD states, the *I* option determines the last ACK state timeout and the *f* option uses FIN\_WAIT\_1 state timeout for fingerprinting.
- (v) *Queso*: This utility runs on Linux and Solaris operating systems. It is used to remotely determine the operating system's version and manufacturer information by analyzing network packets. It provides precise information about a network or a system by scanning the network.
- (vi) *AmapV4.8*: The Amap fingerprinting tool identifies applications and services by creating bogus communication without listening on default ports. It maintains a database of all the known applications, including non-ASCII based applications and enterprise services.
- (vii) *Disco*: The Disco fingerprinting tool is used to discover unique IP addresses on a network. In addition to IP discovery, it also fingerprints TCP SYN packets.
- (viii) *Sprint*: This fingerprinting tool is used to identify the operating system running on a machine. In addition, sprint also has the ability to calculate up times and contains an advanced banner grepping functionality. Sprint, when run with *-n* switch, simulates netcraft.

Among the fingerprinting tools discussed above, Nmap is the best due to its multiple functionalities and superior effectiveness compared to the others. Nmap provides detailed information

Table 6: Comparison of fingerprinting tools

Tool's name	Input	Protocol	Purpose	Effectiveness	Sources
Nmap	H/N	C/U	network scanning	easy to use, powerful, widely used	www.nmap.org
P0f	H	C	remote network identification	passive fingerprinting, difficult to use	www.lcamtuf.coredump.cx
Xprobe	T	C/U	port scanning	simple as nmap, powerful	www.sourceforge.net
CronOS	T	C	OS identification	Linux based	pen-testing.sans.org
Queso	H	C/U	OS fingerprinting	multi-platform, widely used	www.tools.10t3k.net
AmapV4.8	T/p	C/U	host and port scanning	powerful application mapper	www.linux.softpedia.com
Disco		C	IP discovery and Fingerprinting	support large number of features	www.tools.10t3k.net
Sprint	H	C	OS identification	simple and efficient	www.safemode.org

Here, H-Host IP, N-Network IP, T-Target IP, C-TCP, U-UDP, p-Port

on a network or a host with a maximum amount of vulnerable information. P0f supports passive scanning whereas Xprobe and Queso<sup>2</sup> are used for remote operations.

### 3.2.6. User Attack Tools

In user attacks [42], either the attacker (a) attempts as a normal legitimate user to gain the privileges of a root or superuser, or (b) attempts to access a local machine by exploiting its vulnerabilities without having an account on that machine. Both types of attempts are very difficult to detect because their behavior resembles normal characteristics. We discuss these attacks by category along with launching tools.

(i) *U2R Attack*: In this attack, as shown in Figure 5, the attacker initially attempts to gain access to the local victim machine as a legitimate user. The means may be a password sniffing attempt, dictionary attack, or any social engineering approach. The attacker then explores possible vulnerabilities or bugs associated with the operating system running on the victim machine to perform the transition from user to superuser or root level. Once root privileges are acquired, the attacker possesses full control of the victim machine to install backdoor entries for future exploits, manipulate system files to gather information, and other damaging actions. Two well-known U2R attack tools are described next.

(a) *Yaga*: This tool is used to create a new administrator account by compromising registry files. The attacker edits the registry file to crash some system services on the victim machine and create a new administrator account.

(b) *SQLattack*: Here, the attacker creates a TCP connection with an SQL database server on a Unix machine. The database shell exits when a special escape sequence is issued and the root shell of the machine is started by running the Perlmagic<sup>3</sup> script.

(ii) *R2L Attack*: In this attack, a remote attacker, without an account on a local machine, attempts to send packets to that machine by gaining local access based on the vulnerabilities of that machine. To gain access to the local machine, the attacker attempts various ways

<sup>2</sup><http://spot-act.heck.in/queso-scanner-v-0-5.xhtml>

<sup>3</sup><http://www.perlmagic.org/>

as shown in Figure 6. Two such ways are (a) using online and offline dictionary attacks to acquire the password to access the machine, and (b) making repeated guesses at possible usernames and passwords. The attacker also attempts to take advantage of those legitimate users who are often casual in choosing their passwords. Below are two R2L attack tools.

- (a) *Netcat*: This R2L attack tool uses a Trojan program to install and run Netcat on the victim machine at port number 53. The Netcat program works as a backdoor to access the machine using Netcat port without any username and password.
- (b) *ntfsdos*: The attacker gains the console of a WinNT machine by running ntfsdos. The program mounts the machine's disk drives. Thus the attacker is able to copy secret files on the secondary media.

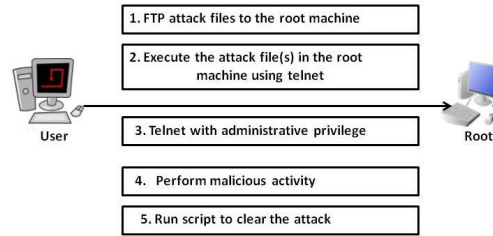


Figure 5: Steps in U2R attack

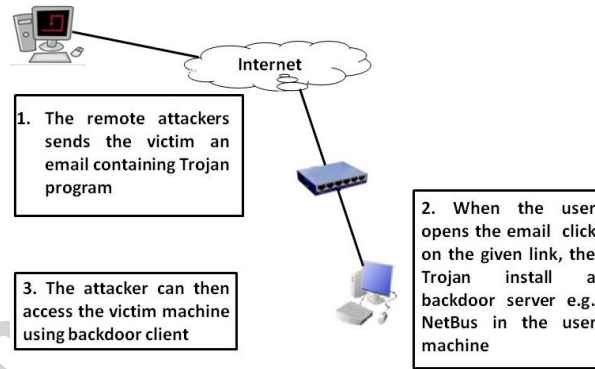


Figure 6: Steps in R2L attack

### 3.2.7. Other Attack Tools

In addition to the tools reported in the preceding sections, there are other tools that have direct or indirect use in the attack launching process. We discuss some of these to increase the awareness of learners and security researchers.

- (i) *Ping*: This tool is used to test network connectivity or reachability of a host on an IP network. Ping is a pioneering tool developed to check a computer or router, and Internet connectivity. The ping request is sent to a particular host or to a network using command prompt. As a reply it displays the response of the destination host and how long it takes



to receive a reply. It uses the ICMP protocol, which has low priority and slower speed than regular network traffic.

- (ii) *Hping2*: It is used to send custom TCP/IP packets and display reply messages received from the target. It handles fragmentation and arbitrary packet size, and can also be used to transfer files. It performs firewall rule testing, port scanning, protocol based network performance testing, and path MTU discovery.
- (iii) *Hping3*: This tool works almost like Hping2 and can handle fragmentation with arbitrary packet size. It finds the sequence number for reply packets from the source port. It starts with a base source port number and increases this number as packets are sent. The default base source port is random. The source port number may be kept constant for each packet sent.
- (iv) *Traceroute*: Traceroute is used to show the route between two systems in a network. It lists all intermediate routers from the source end to the destination end. Using this tool, one determines how systems are connected to each other or how an Internet service provider connects to the Internet to provide services. The traceroute program is available on most computers including most Unix systems, Mac OS and Windows OS.
- (v) *Tctrace*: Though tctrace is similar to traceroute, it uses TCP SYN packets to trace. This makes it possible for one to trace through firewalls if one knows a TCP service that is allowed to pass from the outside.
- (vi) *Tcptraceroute*: Tcptraceroute sends either UDP or ICMP ECHO request packets using a TTL field, which is incremented by one with each hop until the destination is reached. It shows the path that a packet has traversed to reach the destination. However, due to the widespread use of firewall filters it may not be able to complete the path to the destination.
- (vii) *Traceproto*: Traceproto is similar to traceroute, but this tool allows the user to choose protocols to be traced. It currently allows TCP, UDP and ICMP protocol trace. It can be used to test and bypass firewalls, packet filters and check if ports are open. Traceproto is actually a traceroute replacement tool written in C.
- (viii) *Fping*: Fping uses ICMP protocol to determine whether a host is active or not. Fping is more powerful than ping because it can scan any number of hosts or a file containing the list of hosts. Instead of trying one host until it times out or replies, fping sends out a ping packet and moves to the next host in a round-robin fashion. If a host replies, it is noted and removed from the list of hosts to check. If a host does not respond within a certain time limit and/or retry limit, it is considered unreachable. Unlike ping, fping is meant to be used in scripts and its output is easy to parse.
- (ix) *Arping*: The arping tool is used in the Linux platform to send ARP request messages to a destination host in a LAN. It is used to test whether an IP address is in use or not.

Table 7: Comparison of other tools

Tool's name	Input	Protocol	Purpose	Effectiveness	Sources
Ping	H	IC	host discovery	commonly used, easy to use	www.download.cnet.com
Hping2	T	IC/C/U	port scanning	supports many features	www.hping.org.
Hping3	T	IC/C/U	port scanning	powerful for network testing	www.hping.org.
Traceroute	H	IC/C/U	route discovery	multi-platform, easy to use	www.brothersoft.com
Tctrace	H	C	route discovery	multi-platform, easy to use	www.tcptrace.org
Tcptrace route	I	C	DNS lookup	powerful for route discovery	www.michael.toren.net
Traceproto	H	C/U/IC	route discovery	effective for firewall testing	www.traceproto.sourceforge.net
Fping	T	IC	target host discovery	more powerful than ping	www.softpedia.com
Arping	S	A	send ARP request	Linux based, efficient	www.linux.softpedia.com
Here, H-Host IP, T-Target IP, I-Interface ID, S-Source IP, C-TCP, U-UDP, IC-ICMP, A-ARP					

### 3.3. Network Monitoring Tools

Monitoring of network traffic is an essential activity for network defenders in order to observe, analyze and finally identify any anomalies occurring in the network. In support of such activities of network defenders as well as to assist in meaningful interpretation of the outcomes of their analysis, network monitoring and analysis tools play an important role. Rapid incidences of malicious attempts to compromise the confidentiality, integrity and access control mechanisms of a system or to prevent legitimate users of a service from accessing the requested resources have led to an increased demand for developing useful tools to visualize network traffic in a meaningful manner to support subsequent analysis.

#### 3.3.1. Visualization and Analysis tools

An effective network traffic (both packet and NetFlow traffic) visualization tool can be of significant help for network defenders in monitoring and analysis tasks. Appropriate visualization not only supports meaningful interpretation of the analysis results, but also assists security managers in identifying anomalous patterns. It also helps in taking appropriate action to mitigate attacks before they propagate and infect other parts of the network. Some visualization tools are discussed below.

- (i) *Tnv*: This time-based traffic visualization tool presents packet details and links among local and remote hosts. It assists in learning the normal patterns in a network, investigating packet details, and in network troubleshooting. Tnv provides multiple services to support inspection and analysis activities: (a) opening and reading libpcap files, (b) capturing live packets, and (c) saving captured data in a MYSQL database.
- (ii) *Network Traffic Monitor*: This tool provides support in presenting and scanning detailed traffic scenarios since the inception of an application process and also allows analyzing traffic details.
- (iii) *Rumint*: This tool enables visualization of live captured traffic. It can also save captured traffic as a pcap file in the Windows environment.

- (iv) *EtherApe*: EtherApe allows one to sniff live packets and to monitor captured data in the Unix environment.
- (v) *NetGrok*: It is a real-time network visualization tool that presents a graphical layout and a tree map to support visual description of the network data. It can visualize live packets, capture traces, and help in filtering.
- (vi) *NetViewer*: This tool not only supports observation of captured live traffic in an aggregated way, but also helps identify network anomalies. In addition, NetViewer supports visualization of useful traffic characteristics to assist in tuning of defense mechanisms.
- (vii) *VizNet*: It helps visualize the performance of a network based on bandwidth utilization.

Most visualization tools discussed above support both visualization and analysis of network traffic. To visualize and also to analyze a network, one can use EtherApe in Unix or NetViewer in

Table 8: Comparison of visualization tools

Tool's name	Input	Protocol	Purpose	Effectiveness	Sources
Tnv	F	C/U/ IC	traffic visualization	supported by all OSs	<a href="http://www.tnv.sourceforge.net">www.tnv.sourceforge.net</a>
Network Traffic Monitor 1.02	H	C/U/ IC	live traffic monitoring	easy to use, efficient for visualization	<a href="http://www.monitor-network-traffic.winsite.com">www.monitor-network-traffic.winsite.com</a>
Rumint	H	C/U/ IC/IG	visualize life traffic	extremely flexible	<a href="http://www.rumint.org">www.rumint.org</a>
EtherApe	H	C	traffic flow visualization	very simple and powerful	<a href="http://www.brothersoft.com">www.brothersoft.com</a>
Netgrok	H	C/U/ IC	real-time traffic visualization	multi-platform and easy to use	<a href="http://www.softpedia.com">www.softpedia.com</a>
Netviewer	H	C/U	traffic analysis	powerful defense tool	<a href="http://www.brothersoft.com">www.brothersoft.com</a>
VizNet	H	C/U	traffic analysis and visualization	efficient for visualization	<a href="http://www.viznet.ac.uk">www.viznet.ac.uk</a>
Here, H-Host IP, F-Captured data file, C-TCP, U-UDP, IC-ICMP, IG-IGMP					

the Windows platform. For real-time visualization of live traffic for intrusion detection, NetViewer is the best due to its ability to detect anomalous network traffic. Network defenders need real-time visualization tools that can detect abnormal behaviors in network traffic and immediately generate alert messages to inform the administrator.

#### 4. Attack Detection Systems

Attack detection systems or intrusion detection systems are essential to keep enterprise networks secure. Many IDSs have been developed with diverse facilities. However, there is still need for a complete real-time solution with novel attack detection capability. We describe here some popular IDSs with architecture for a selected few. We also present a comparison among them. The intrusion detection systems that we discuss attempt to identify known as well as unknown attacks using statistical, data mining or soft computing approaches.

- (i) ADAM [43]: Automated Data Analysis and Mining (ADAM) includes a data mining based technique to detect intrusions or attacks within a testbed. It combines association rule

Table 9: Tools by Category

Category	Tool name	Source
Trojans	NukeNabber	<a href="http://community.norton.com">http://community.norton.com</a>
	AIMSpy	<a href="http://www.securitystronghold.com">http://www.securitystronghold.com</a>
	NetSpy	<a href="http://www.netspy-trojan-horse.downloads">http://www.netspy-trojan-horse.downloads</a>
Information Gathering Tools	ASS	<a href="http://www.manpages.ubuntu.com">http://www.manpages.ubuntu.com</a>
	NMap	<a href="http://www.nmap.org">http://www.nmap.org</a>
	p0f	<a href="http://www.lcamtuf.coredump.cx/p0f.shtml">http://www.lcamtuf.coredump.cx/p0f.shtml</a>
	MingSweeper	<a href="http://www.hoobie.net/mingsweeper">http://www.hoobie.net/mingsweeper</a>
	THC Amap	<a href="http://www.freeworld.thc.org/thc-amap">http://www.freeworld.thc.org/thc-amap</a>
DoS attack tools	Angry IP Scanner	<a href="http://www.angryziber.com/w/Download">http://www.angryziber.com/w/Download</a>
	Targa	<a href="http://www.security-science.com/">http://www.security-science.com/</a>
	Burbonic	<a href="http://www.softpedia.com">http://www.softpedia.com</a>
Spoofing attack tools	Blast20	<a href="http://seomagz.com/2010/03/dos-denial-of-service-attack-tools-ethical-hacking-session-3/">http://seomagz.com/2010/03/dos-denial-of-service-attack-tools-ethical-hacking-session-3/</a>
	Engage Packet Builder	<a href="http://www.engage-packet-builder.software.informer.com/">http://www.engage-packet-builder.software.informer.com/</a>
	Hping	<a href="http://www.hping.org">http://www.hping.org</a>
	Nemesis	<a href="http://www.nemesis.sourceforge.net">http://www.nemesis.sourceforge.net</a>
	PacketExcalibur	<a href="http://www.linux.softpedia.com">http://www.linux.softpedia.com</a>
TCP Session Hijacking Tools	Scapy	<a href="http://www.softpedia.com">http://www.softpedia.com</a>
	Firesheep	<a href="http://www.codebutler.github.com/firesheep/">http://www.codebutler.github.com/firesheep/</a>
	Hunt	<a href="http://www.packetstormsecurity.org/sniffers/hunt">http://www.packetstormsecurity.org/sniffers/hunt</a>
	Juggernaut	<a href="http://www.tools.l0t3k.net/Hijacking/1.2.tar.gz">http://www.tools.l0t3k.net/Hijacking/1.2.tar.gz</a>
	TTY Watcher	<a href="http://www.security-science.com">http://www.security-science.com</a>
Probe Attack Tools	IP Watcher	<a href="http://www.download.cnet.com">http://www.download.cnet.com</a>
	Hjksuit-v0.1.99	<a href="http://www.tools.l0t3k.net/Hijacking/hjksuite-0.1.99.tar.gz">http://www.tools.l0t3k.net/Hijacking/hjksuite-0.1.99.tar.gz</a>
	Solarwind	<a href="http://www.solarwinds.com">http://www.solarwinds.com</a>
Spoofing Attack Tools in Wireless	Network Probe	<a href="http://www.softpedia.com">http://www.softpedia.com</a>
	NMap	<a href="http://nmap.org">http://nmap.org</a>
	Kismet	<a href="http://www.linux.die.ne">http://www.linux.die.ne</a>
	libpcap	<a href="http://www.sourceforge.net/projects/libpcap">http://www.sourceforge.net/projects/libpcap</a>
	libnet	<a href="http://www.libnet.sourceforge.net">http://www.libnet.sourceforge.net</a>
Application Layer Attack Tools	libdnet	<a href="http://www.libdnet.sourceforge.net/">http://www.libdnet.sourceforge.net/</a>
	libradiate	<a href="http://www.packetfactory.net/projects/libradiate">http://www.packetfactory.net/projects/libradiate</a>
	HOIC	<a href="https://www.rapidshare.com">https://www.rapidshare.com</a>
	LOIC	<a href="http://www.softpedia.com">http://www.softpedia.com</a>
	RefRef	<a href="http://www.softpedia.com">http://www.softpedia.com</a>

mining and classification to discover attacks in tcpdump audit trail data. ADAM trains the classifier to classify suspicious connections as either a known type of attack or an unknown type or a false alarm with respect to the existing rules or profiles.

- (ii) MINDS [44]: Minnesota Intrusion Detection System (MINDS) is another popular data mining based system for detecting network attacks or intrusions. The architecture of MINDS is shown in Figure 7. It takes NetFlow version 5 data collected through flow tools. Before entering the anomaly detection module, a data filtering step is executed to remove non-interesting network traffic patterns. The anomaly detection engine uses an outlier detection algorithm to assign an anomaly score to each network connection. Finally, it reports alarms for any malicious activity based on the anomaly scores.
- (iii) DNIDS [45]: The Dependable Network Intrusion Detection System (DNIDS) uses a combined strangeness and isolation measure with the k-nearest neighbor (CSI-KNN) algorithm. DNIDS can detect network intrusions while providing continued service even under attacks. The intrusion detection algorithm analyzes characteristics of network data by employing two measures, strangeness and isolation. Based on these measures, a correlation unit raises intrusion alerts with associated confidence estimates. Multiple CSI-KNN classifiers work in parallel to deal with different types of network traffic and report alarm for any abnormal traffic patterns.
- (iv) HIDE [46]: HIDE is a hierarchical anomaly based system, developed using statistical mod-

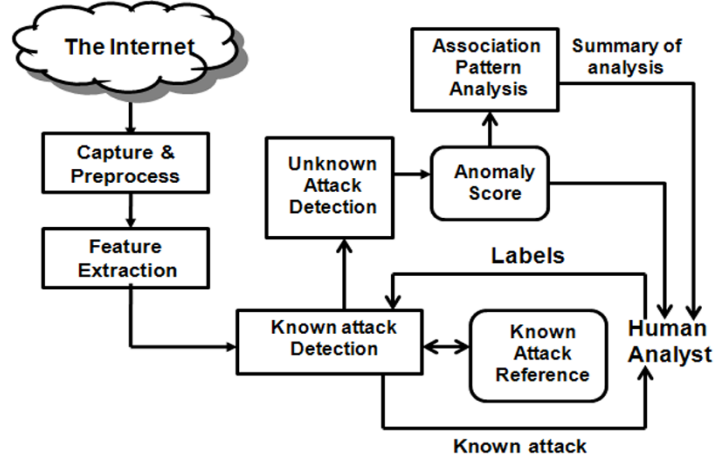


Figure 7: Architecture of MINDS

eling and neural networks. See Figure 8 for architecture of the system. It consists of several tiers, each tier containing several Intrusion Detection Agents (IDAs), which are IDS components that monitor activities of a host or a network. The statistical processor maintains a reference model of typical network activities, compares reports from the event preprocessor with the reference model. It forms a stimulus vector to feed into the neural network classifier that analyzes the vector to decide whether the network traffic is normal or attack. The post-processor generates reports for agents at higher tiers.

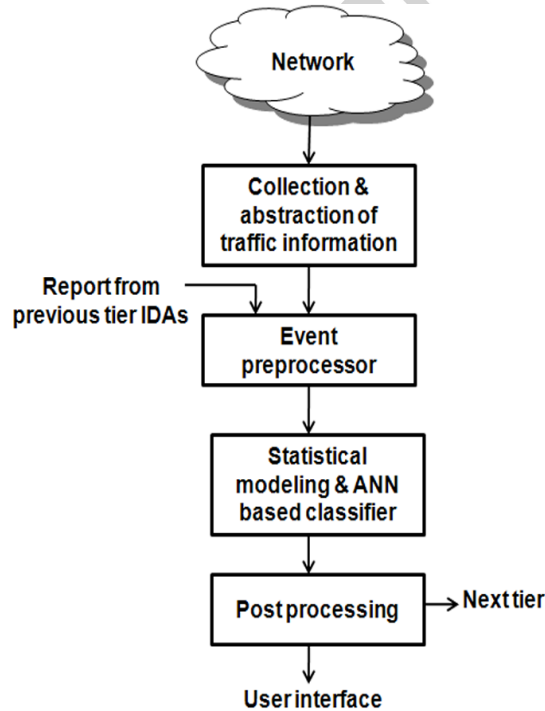


Figure 8: Architecture of HIDE

- (v) NSOM [47]: Network Self-Organizing Maps (NSOM) is a self-organizing map (SOM) based IDS that attempts to detect anomalies by quantifying the usual or acceptable behavior and

by flagging irregular behavior as potentially intrusive. NSOM is used to classify real-time Ethernet network traffic data. It collects network traffic data continuously from a network port, preprocesses and selects suitable features to classify them as attack or normal.

- (vi) FSAS [48]: Flow-based Statistical Aggregation Scheme for Network Anomaly Detection (FSAS) has a two-layered architecture containing a feature generator and a flow-based detector. The feature generator collects network traffic data from a host or a network and the event time module periodically calls the feature extraction module to convert the flow statistic information into the format required by the model. The feature scoring metric calculates the probability scores of these features by comparing the features with the reference model generated by past normal and attack users. Higher the maliciousness of a flow, the higher is the possibility of the flow being an attack. FSAS provides 22 significant features relevant for DoS attack detection.
- (vii) N@G [49]: Network at Guard (N@G) is a hybrid IDS that contains both network and host sensors. It analyzes the audit trail using statistical techniques as part of the host sensor. The system has a management console to aggregate alerts from various sensors using a user interface, a middle tier and a data management component. It provides real-time protection to client components against malicious traffic, which can include unsolicited changes to the Windows hosts file, and Windows messenger service. It also provides Layered Service Provider (LSP) and Domain Name Server (DNS) protection. The system can dynamically apply access control to routers (Cisco) to actively block network attacks.
- (viii) FIRE [50]: Fuzzy Intrusion Recognition Engine (FIRE) is a fuzzy logic based anomaly IDS to assess malicious activity. The system combines simple network traffic metrics with fuzzy rules to determine the likelihood of specific or general network attacks. FIRE relies on fuzzy network traffic profiles as inputs to its rule set and uses simple data mining techniques to process the network input data for anomaly detection.
- (ix) NFIDS [51]: NFIDS is a hierarchical neuro-fuzzy anomaly IDS that is composed of several autonomous agents and three tiers. Tier-I contains some Intrusion Detection Agents (IDAs) that monitor activities of a host or a network and report abnormal behavior to Tier-II. Tier-III combines correlation data, higher-level reports and sends alarm to the user interface. This system uses a decision making process to detect intrusions based on fuzzy rules and neural networks. Fuzzy rules are created using expert knowledge of system administrators to represent common types of attacks. Fuzzy rules can be learned by the neural network structure and based on the membership values of the fuzzy rules, different attacks are detected.
- (x) D-WARD [52]: D-WARD is an adaptive source-end DDoS defense system that can detect attacks autonomously and give surgically accurate response using its traffic profiling techniques. The architecture of D-WARD is shown in Figure 9. D-WARD inflicts very low

collateral damage to legitimate traffic, while quickly detecting and severely rate-limiting outgoing attacks.

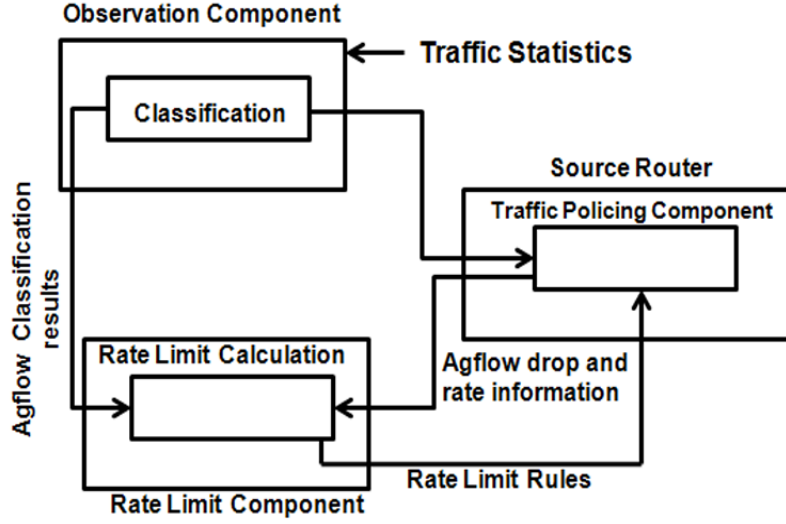


Figure 9: Architecture of D-WARD

- (xi) LADS [53]: Large-scale Automated DDoS detection System (LADS) is a triggered multi-stage detection system that addresses both scalability and accuracy in detecting DDoS attacks. LADS uses clustering on NetFlow data to detect DDoS attacks in a Tier-1 ISP.
- (xii) ANTID [54]: ANTID detects and filters DDoS attacks which use spoofed packets to circumvent conventional intrusion detection schemes. The anti-DDoS scheme intends to complement, rather than replace conventional schemes by embedding in each IP packet a unique path fingerprint that represents the route an IP packet has traversed. Thus, ANTID is able to distinguish IP packets that traverse different Internet paths. A spoofed DDoS attack can be detected by observing a surge of spoofed packets. ANTID is lightweight, robust, and incrementally deployable.
- (xiii) DCD [55]: DCD uses Change Aggregation Trees (CAT) to detect distributed flooding attacks at flow-level. The idea is to detect abrupt traffic changes across multiple network domains at the earliest time. The system is built over attack-transit routers, which work together. Each ISP domain has a CAT server to aggregate flooding alerts reported by routers. To resolve policy conflicts at different ISP domains, a new secure infrastructure protocol (SIP) was developed to establish mutual trust or consensus.
- (xiv) CERN Investigation of Network Behavior and Anomaly Detection (CINBDS) [56]: The main objective of this project is to detect anomalies in a network and automatically take countermeasures. It works in high speed networks. It uses the sFlow [14] standard for monitoring a high speed network and reduce occurrences of false positive alarms. However, for unknown anomalies this system generates false positives.

- (xv) NetSTAT [57]: It is a network based intrusion detection system that uses the state transition analysis technique. It operates on a high volume network. State transition analysis describes computer penetrations as a sequence of actions performed by an attacker to compromise a system. This system uses autonomous intrusion detection components known as probes. Probes are used to monitor network traffic and a filter module is used to select the messages that contribute assertions in a state transition scenario. If a single probe can detect all types of attacks, it does not interact with the analyzer. Otherwise, the analyzer decomposes an intrusion scenario into subscenarios so that each one can be detected by a single probe.
- (xvi) Bro [58]: It is a real-time stand-alone system used for detecting network intruders. Bro is divided into an event engine that reduces a kernel filtered network traffic stream into a series of high level events, and a policy script interpreter that interprets event handlers written in a specialized language used to express a site's security policy. It detects intrusions by parsing network traffic to extract application level semantics. The Bro analyzer filters network traffic and removes unnecessary elements. The remaining information is sent to the event engine and Bro interprets the structure of network packets and abstracts them into high-level events that are analyzed by policy script interpreter to detect malicious activities.
- (xvii) Snort [59]: Snort is a cross-platform, lightweight network intrusion detection system that can detect a wide variety of traffic anomalies over TCP/IP networks. It is useful when it is not cost efficient to deploy a commercial NIDS. Though Snort operates in a manner similar to tcpdump, it can inspect packet payload information. It decodes the application layer packets and uses rules to collect traffic that has specific patterns in the application layer. The decoded output produced by Snort is more user friendly than tcpdump. To detect network anomalies, Snort uses rules and matching algorithms.
- (xviii) PAYL [60]: It is a payload based intrusion detection system that uses statistical measures to detect anomaly patterns. It makes a profile of byte frequency distribution and the standard deviation of payloads to a single host or port for normal data. During anomaly detection, it captures incoming payloads and computes the Mahalanobis distance metric from the normal. Any new test payload found to be too distant from the normally expected payload is deemed anomalous and an alert is generated.
- (xix) ALERT-ID [61]: It is a network based intrusion detection system that oversees activities of a network and generates an alarm on detecting malicious patterns. It detects anomaly based on comparison of real-time captured traffic from switch/router logs and profiles constructed from historical data. This system effectively identifies potential intrusions and misuses with an acceptable overall alarm rate.
- (xx) MAD-IDS [62]: MAD-IDS is a distributed intrusion detection system that exploits the features obtained from mobile agent methodology. It depends on (a) a misuse detection mobile agent to detect known attacks and (b) an anomaly detection mobile agent to detect



unknown attacks. The mobile agents provide high accuracy for predicting different behaviors in network traffic using the data mining techniques.

- (xxi) ML-DIDS [63]: It is a signature-based multi-layer distributed IDS that uses mobile agents. It can detect threats by dynamically creating efficient multiple small databases for signatures. At the same time, it provides a mechanism to update these multiple small signature databases at regular intervals using mobile agents with high detection rate.

A comparison of detection systems based on parameters such as detection type (host based, network based or both), detection approach (misuse, anomaly or both), nature of detection (online or offline), nature of processing (centralized or distributed), data gathering mechanism (centralized or distributed) and the technical approach for analysis, is given in Table 10. We make the following observations regarding attack detection systems.

- A detection system with high accuracy and low false alarm rate may often fail to operate in real-time. However, if one compromises on the false alarm rate, one can build effective real-time systems.
- A detection system should be capable of not only detecting unknown attacks, but also be able to modify the normal as well as attack profiles dynamically.

Table 10: Comparison of attack detection systems, where P represents the types of detection as host based (H) or network based (Net) or hybrid (H), W indicates the class of detection as misuse (M) or anomaly (A) or both (B), X corresponds to the nature of detection as real-time (R) or non-real time (N), Y and Z represent the nature of processing and data gathering mechanism as centralized (C) or distributed (D) respectively.

System	Year	P	W	X	Y	Z	Approach
NetSTAT [57]	1999	Net	A	R	C	D	State transition approach
Bro [58]	1999	Net	A	R	C	C	Behavior analysis
Snort [59]	1999	Net	A	R	C	D	Rule based
FIRE [50]	2000	Net	A	N	C	C	Fuzzy Logic
ADAM [43]	2001	Net	A	R	C	C	Association Rule
HIDE [46]	2001	Net	A	R	C	D	Statistical and Neural Network
NSOM [47]	2002	Net	A	R	C	C	Neural Networks
MINDS [44]	2003	Net	A	R	C	C	Outlier
NFIDS [51]	2003	Net	A	N	C	C	Neuro Fuzzy Logic
N@G [49]	2003	H	B	R	C	C	Statistical
PAYL [60]	2004	H	A	R	C	C	Statistical
D-WARD [52]	2005	Net	B	R	D	D	Statistical
ANTID [54]	2005	Net	A	R	C	D	Path fingerprinting
FSAS [48]	2006	Net	A	R	C	C	Statistical
LADS [53]	2006	Net	A	R	C	D	Clustering based
DNIDS [45]	2007	Net	A	R	C	C	K-NN
DCD [55]	2007	Net	A	R	C	D	Statistical
MAD-IDS [62]	2010	Net	B	R	D	D	Mobile agent based
ALERT-ID [61]	2012	Net	A	R	C	D	Rule based
ML-DIDS [63]	2013	Net	M	R	D	D	Misuse mobile agent

## 5. Observations and Conclusions

Based on our extended study of network security tools and systems available for anomaly based NIDS, we make the following observations.

- Existing information gathering tools that scan the network work successfully in one-to-one and one-to many scenarios. However, existing tools are unsuitable for coordinated scanning (i.e., m:1 and n:m mapping) with varying source and destination IPs, dynamically within a specified time interval.
- An integrated tool with supporting modules for capture, preprocessing, analysis, and visualization of both packet and NetFlow data is lacking. Existing tools (e.g., wireshark, Nfsen, and Nfdump) can support capture of either NetFlow traffic or packet label traffic but not both.
- Existing DDoS attack tools cannot launch attacks at multiple layers. They support launching of only single layer attacks.
- Most existing DDoS attack tools are restricted to a limited number of attack scenarios. Such tools cannot be customized to develop additional attack scenarios.
- Most existing NIDSs are dependent on several user input parameters, and their performance is highly sensitive to these parameters.
- Almost all anomaly based NIDSs perform either near real-time or offline. In addition, most suffer from a large number of false alarms.
- An effective tool to support correlation between packet traffic and NetFlow traffic is still lacking.

Based on above observations, we have identified the following list of research challenges for network security researchers.

- It is a challenging task to develop an integrated tool to support capture, preprocessing (e.g., filtering, and feature extraction), analysis, and visualization of both packet and NetFlow traffic.
- It is also a challenging task to develop a tool for faster capture, preprocessing and extraction of all types features for network traffic corresponding to all layer protocols.
- It is a non-trivial task to develop a GUI based DDoS attack traffic generation tool that is capable of handling all possible attack scenarios for multiple layers (e.g., application and transport layers).
- Development of an anomaly based NIDS is dependent on a minimum number of user parameters and capable of handling both known as well as unknown attacks in real-time with a minimum number of false alarms is another challenging task.
- Development of a real-time detection system for both low rate and high rate DDoS attacks at the victim end without affecting legitimate users or normal service is another challenging task.

Even though there are many network security tools available in the research community, the proper use of these tools is very important in the real network security infrastructure. In this paper, we have presented three major categories of security tools depicted in Figure 2. We started this paper with a brief description of network attacks, their characteristics and steps to perform attacks. The paper also presents a large number of tools that have become popular in recent years. We also provide architectures of some popular IDSs and compare them. Finally, we conclude with a list of observations and research challenges.

### Acknowledgment

This work is supported by Department of Information Technology (DIT) and Council of Scientific & Industrial Research (CSIR), Government of India. The research is also partially funded by the National Science Foundation (NSF), USA under grants CNS-095876 and CNS-0851783. The authors are thankful to the funding agencies.

- [1] A. Pras, A. Sperotto, G. C. M. Moura, I. Drago, R. Barbosa, R. Sadre, R. Schmidt, R. Hofstede, Attacks by “anonymous” Wikileaks proponents not anonymous, Tech. Rep. 10.41, Design and Analysis of Communication Systems Group (DACS), University of Twente, Enschede, The Netherlands (December 10 2010).
- [2] S. Mansfield-Devine, Anonymous: serious threat or mere annoyance?, *Network Security* 2011 (1) (2011) 4–10.
- [3] A. Orebaugh, G. Ramirez, J. Beale, Wireshark & Ethereal network protocol analyzer toolkit, Syngress, 2006.
- [4] C. Satten, Lossless gigabit remote packet capture with linux (2007).
- [5] L. Deri, R. Carbone, S. Suin, Monitoring networks using ntop, in: Integrated Network Management Proceedings, 2001 IEEE/IFIP International Symposium on, Seattle, WA, USA, IEEE, 2001, pp. 199–212.
- [6] G. Conti, K. Abdullah, Passive visual fingerprinting of network attack tools, in: *Proceedings of the 2004 workshop on Visualization and Data Mining for Computer Security*, Washington, DC, USA, ACM, 2004, pp. 45–54.
- [7] R. Barber, Hacking techniques: The tools that hackers use, and how they are evolving to become more sophisticated., *Computer Fraud & Security* 2001 (3) (2001) 9–12.
- [8] E. S. Pilli, R. Joshi, R. Niyogi, Network forensic frameworks: Survey and research challenges, *Digital Investigation* 7 (1) (2010) 14–27.
- [9] P. Gogoi, D. Bhattacharyya, B. Borah, J. K. Kalita, A survey of outlier detection methods in network anomaly identification, *Comput. J.* 54 (4) (2011) 570–588.

- [10] C. V. Zhou, C. Leckie, S. Karunasekera, A survey of coordinated attacks and collaborative intrusion detection, *Computers & Security* 29 (1) (2010) 124–140.
- [11] T. F. Lunt, A survey of intrusion detection techniques, *Computers & Security* 12 (4) (1993) 405–418.
- [12] T. Peng, C. Leckie, K. Ramamohanarao, Survey of network-based defense mechanisms countering the DoS and DDoS problems, *ACM Computing Surveys (CSUR)* 39 (1) (2007) 3.
- [13] N. Dhanjani, J. Clarke, Network security tools, O'Reilly Media, USA, 2005.
- [14] B. Li, J. Springer, G. Bebis, M. Hadi Gunes, A survey of network flow applications, *Journal of Network and Computer Applications* 36 (2) (2013) 567–581.
- [15] P. Garcia-Teodoro, J. Diaz-Verdejo, G. Maciá-Fernández, E. Vázquez, Anomaly-based network intrusion detection: Techniques, systems and challenges, *Computers & Security* 28 (1) (2009) 18–28.
- [16] I. Corona, G. Giacinto, F. Roli, Adversarial attacks against intrusion detection systems: Taxonomy, solutions and open issues, *Information Sciences* 239 (2013) 201–225.
- [17] S. Axelsson, Intrusion detection systems: A survey and taxonomy, Tech. rep., Chalmers Univ. (2000).
- [18] H. Debar, M. Dacier, A. Wespi, Towards a taxonomy of intrusion-detection systems, *Computer Networks* 31 (8) (1999) 805–822.
- [19] J. S. Sherif, T. G. Dearmond, Intrusion detection: systems and models, in: *Proceedings of the 11th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises*, Rome, Italy, IEEE, 2002, pp. 115–133.
- [20] A. Lazarevic, V. Kumar, J. Srivastava, Intrusion detection: A survey, in: *Managing Cyber Threats*, Springer, 2005, pp. 19–78.
- [21] V. Chandola, A. Banerjee, V. Kumar, Anomaly detection: A survey, *ACM Computing Surveys* 41 (3) (2009) 15.
- [22] M. Bhuyan, D. Bhattacharyya, J. Kalita, Network Anomaly Detection: Methods, Systems and Tools, *IEEE Communications Surveys and Tutorials Early Access* (2013) 1–34.
- [23] M. H. Bhuyan, D. K. Bhattacharyya, J. K. Kalita, *Surveying Port Scans and Their Detection Methodologies*, The Computer Journal 54 (2011) 1565–1581.
- [24] L. Danielle, Introduction to dsniiff, in: *Global Information Assurance Certification Paper*, SANS Institute, 2002.

- [25] K. H. Yeung, D. Fung, K. Y. Wong, Tools for attacking layer 2 network infrastructure, Vol. 2, *Proceedings of the International MultiConference of Engineers and Computer Scientists*, Hong Kong, 2008, pp. 1–6.
- [26] D. Norton, An Ettercap primer, SANS Institute InfoSec Reading Room, 2004.
- [27] M. D. Schiffman, Libnet 101, part 1: The primer, Guardent security digital infrastructure, 2000, pp. 1–10.
- [28] N. Ye, T. Ehiabor, Y. Zhang, First-order versus high-order stochastic models for computer intrusion detection, *Quality and Reliable Engineering International* 18 (3) (2002) 243–250.
- [29] W.-H. Chen, S.-H. Hsu, H.-P. Shen, Application of SVM and ANN for intrusion detection, *Computer and Operation Research* 32 (10) (2005) 2617–2634.
- [30] F. Jemili, M. Zaghdoud, M. Ben Ahmed, A framework for an adaptive intrusion detection system using bayesian network, in: *Proceedings of the IEEE Intelligence and Security Informatics*, 2007, pp. 66–70.
- [31] H. Debar, M. Dacier, A. Wespi, Towards a taxonomy of intrusion-detection systems, *Computer Networks* 31 (9) (1999) 805–822.
- [32] M. Aydın, A. Zaim, K. Ceylan, A hybrid intrusion detection system design for computer network security, *Computers & Electrical Engineering* 35 (3) (2009) 517–526.
- [33] M. Bhuyan, D. Bhattacharyya, J. Kalita, NADO: network anomaly detection using outlier approach, in: *Proceedings of the 1st International Conference on Communication, Computing & Security*, ACM, New York, NY, USA, 2011, pp. 531–536.
- [34] M. H. Bhuyan, D. K. Bhattacharyya, J. K. Kalita, Survey on incremental approaches for network anomaly detection, *International Journal of Communication Networks and Information Security* 3 (3) (2011) 226–239.
- [35] Y. Xie, S. Z. Yu, Monitoring the application-layer DDoS attacks for popular websites, *IEEE/ACM Transactions on Networking* 17 (1) (2009) 15–25.
- [36] S. Ranjan, R. Swaminathan, M. Uysal, E. Knightly, DDoS-Resilient Scheduling to Counter Application Layer Attacks Under Imperfect Detection, in: *Proceedings of the 25th IEEE International Conference on Computer Communications*, Barcelona, Spain, 2006, pp. 1–13.
- [37] J. Yu, Z. Li, H. Chen, X. Chen, A detection and offense mechanism to defend against application layer DDoS attacks, in: *Proceedings of the 3rd International Conference on Networking and Services*, IEEE, National University of Defense Technology, Changsha, 2007, pp. 54–60.
- [38] R. Beverly, A robust classifier for passive TCP/IP fingerprinting, *Passive and Active Network Measurement* (2004) 158–167.

- [39] S. Shah, An introduction to HTTP fingerprinting, *Net-Square Solutions* (2004) 1–21.
- [40] F. Yarochkin, Remote OS detection via TCP/IP stack fingerprinting, *Phrack Magazine* 17 (3) (1998) 1–10.
- [41] S. Singh, C. Estan, G. Varghese, S. Savage, Automated worm fingerprinting, in: *Proceedings of the 6th Symposium on Operating Systems Design & Implementation - Volume 6*, USENIX Association, Berkeley, CA, USA, 2004, pp. 4–4.
- [42] R. P. Lippmann, R. K. Cunningham, Improving intrusion detection performance using keyword selection and neural networks, *Computer Networks* 34 (4) (2000) 597–603.
- [43] B. Daniel, C. Julia, J. Sushil, W. Ningning, ADAM: a testbed for exploring the use of data mining in intrusion detection, *ACM SIGMOD Record* 30 (4) (2001) 15–24.
- [44] L. Ertoz, E. Eilertson, A. Lazarevic, P. Tan, V. Kumar, J. Srivastava, P. Dokas, MINDS-Minnesota Intrusion Detection System, *Next Generation Data Mining* (2004) 199–218.
- [45] L. V. Kuang, DNIDS: a dependable network intrusion detection system using the CSI-KNN algorithm, Master's thesis, Queen's University Kingston, Ontario, Canada (Sep 2007).
- [46] Z. Zhang, J. Li, C. Manikopoulos, J. Jorgenson, J. Ucles, HIDE: a hierarchical network intrusion detection system using statistical preprocessing and neural network classification, in: *Proceedings of the 2nd Annual IEEE Systems, Cybernetics Information Assurance Workshop*, IEEE Computer Society, West Point, NY, USA, 2001, pp. 85–90.
- [47] K. Labib, R. Vemuri, NSOM: a tool to detect denial of service attacks using self-organizing maps, Tech. rep., Department of Applied Science University of California, Davis, California, U.S.A. (2002).
- [48] S. Song, L. Ling, C. Manikopoulo, Flow-based statistical aggregation schemes for network anomaly detection, in: *Proceedings of the IEEE International Conference on Networking, Sensing and Control*, IEEE, Ft. Lauderdale, Florida, USA, 2006, pp. 786–791.
- [49] N. Subramoniam, P. S. Pawar, M. Bhatnagar, N. S. Khedekar, S. Guntupalli, N. Satyanarayana, V. A. Vijayakumar, P. K. Ampatt, R. Ranjan, P. S. Pandit, Development of a comprehensive intrusion detection system - challenges and approaches, in: *Proc. of the 1st International Conference on Information Systems Security*, Kolkata, India, 2005, pp. 332–335.
- [50] J. E. Dickerson, Fuzzy network profiling for intrusion detection, in: *Proc. of the 19th International Conference of the North American Fuzzy Information Processing Society*, Atlanta, 2000, pp. 301–306.

- [51] M. Mohajerani, A. Moeini, M. Kianie, NFIDS: a neuro-fuzzy intrusion detection system, in: *Proc. of the 10th IEEE International Conference on Electronics, Circuits and Systems* at the University of Sharjah in Sharjah, United Arab Emirates, Vol. 1, 2003, pp. 348–351.
- [52] J. Mirkovic, P. Reiher, D-ward: A source-end defense against flooding denial-of-service attacks, *IEEE Transactions on Dependable Secure Computing* 2 (3) (2005) 216–232.
- [53] V. Sekar, N. Duffield, O. Spatscheck, J. van der Merwe, H. Zhang, LADS: large-scale automated DDoS detection system, in: *Proceedings of the annual conference on USENIX '06 Annual Technical Conference*, USENIX Association, Berkeley, CA, USA, 2006, pp. 16–16.
- [54] F.-Y. Lee, S.-P. Shieh, Defending against spoofed DDoS attacks with path fingerprint, *Computers & Security* 24 (2005) 571–586.
- [55] Y. Chen, K. Hwang, W.-S. Ku., Collaborative detection of DDoS attacks over multiple network domains, *IEEE Transactions on Parallel Distributed Systems* 18 (12) (2007) 1649–1662.
- [56] M. M. Hulboj, R. E. Jurga, CERN investigation of network behaviour and anomaly detection, in: *Recent Advances in Intrusion Detection*, Springer Berlin Heidelberg, 2009, pp. 353–354.
- [57] G. Vigna, R. A. Kemmerer, Netstat: A network-based intrusion detection system, *Journal of Computer Security* 7 (1) (1999) 37–71.
- [58] V. Paxson, Bro: a system for detecting network intruders in real-time, *Computer Networks* 31 (23) (1999) 2435–2463.
- [59] M. Roesch, Snort - lightweight intrusion detection for networks, in: *Proc. of the 13th USENIX Conference on System Administration*, Washington, 1999, pp. 229–238.
- [60] K. Wang, S. J. Stolfo, Anomalous payload-based network intrusion detection, in: *Recent Advances in Intrusion Detection*, Springer, 2004, pp. 203–222.
- [61] J. Chu, Z. Ge, R. Huber, P. Ji, J. Yates, Y.-C. Yu, Alert-ID: analyze logs of the network element in real time for intrusion detection, in: *Research in Attacks, Intrusions, and Defenses*, Springer, 2012, pp. 294–313.
- [62] I. Brahmi, S. B. Yahia, P. Poncelet, MAD-IDS: novel intrusion detection system using mobile agents and data mining approaches, in: *Proceedings of the Pacific Asia conference on Intelligence and Security Informatics*, Springer-Verlag, Berlin, Heidelberg, 2010, pp. 73–76.
- [63] M. Uddin, A. A. Rehman, N. Uddin, J. Memon, R. Alsaqour, S. Kazi, Signature-based Multi-Layer Distributed Intrusion Detection System using Mobile Agents, *International Journal of Network Security* 15 (1) (2013) 79–87.